Deep Learning Methods for Real-Time Speech & Audio

Bandhav Veluri

A dissertation

submitted in partial fulfillment of the

requirements for the degree of

Doctor of Philosophy

University of Washington
2025

Reading Committee:
Shyam Gollakota, Co-Chair
Michael Bedford Taylor, Co-Chair
Steven M Seitz
Luke Zettlemoyer

Program Authorized to Offer Degree:
Computer Science and Engineering

University of Washington

Abstract

Deep Learning Methods for Real-Time Speech & Audio

Bandhav Veluri

Co-chairs of the Supervisory Committee:

Shyam Gollakota
Computer Science and Engineering

Michael Bedford Taylor
Computer Science and Engineering

Deep learning has revolutionized speech and audio processing, bringing a step change in the state-of-the-art for tasks such as speech recognition, speech separation, and audio event detection. However, these developments have mainly focused on offline processing of speech and audio, with limited emphasis on applications involving very low lookahead waveform-to-waveform transformations. My research tackles two classes of such problems on opposite ends of the spectrum regarding latency requirements and problem complexity.

At the lower end of latency tolerance, we propose *semantic hearing*, a set of methods for semantically customizing one's perceived acoustic environment, requiring signal-level understanding with a latency as low as 20 milliseconds between an output audio chunk and the corresponding input chunk. On the other end, we developed spoken language models that enable full-duplex voice interaction, requiring human-level understanding of real-time spoken dialogues. Furthermore, we investigate modeling with raw acoustic representations of input speech, in contrast to prevalent speech representations referred to as semantic units. We show that understanding acoustic representations improves the robustness of spoken dialogue models in noisy scenarios where interfering speech is present.

Acknowledgements

I'm deeply grateful to my advisors, Shyam Gollakota and Michael Taylor, for giving me the opportunity to do my PhD with them. Their guidance and support were so important throughout this journey. I vividly recall an early conversation with Shyam about the power of thinking from a systems perspective for bringing AI to life in everyday devices. His insight into the symbiosis between system design and model design immediately resonated with me and became a core theme of my work.

This theme showed up directly in much of my PhD research: system constraints guided algorithm design in projects like LookOnceToHear and SyncLLM, and algorithmic advances led to new applications like Semantic Hearing. I was also lucky to explore these ideas further through many lively, open-ended group discussions led by Shyam.

But none of this might have happened without Michael's early trust in me, even as an undergraduate, giving me the chance to co-develop the 511-core manycore processor with him. I'm incredibly grateful for our continued collaboration during my master's, especially in developing the PyTorch backend for that manycore architecture. These early experiences were invaluable; they gave me key insights into the complexities of ML systems and truly marked my foray into deep learning, laying the foundation for my PhD research.

I thank my supervisory committee, Anind Dey, Steve Seitz, Luke Zettlemoyer and Hongyu Gong. Anind's extensive research in human-computer interaction is inspiring and I aspire to developing such widely useful technologies. Steve's exciting works on audio processing, The Cone of Silence and ClearBuds, have proposed methods that are directly relevant to my own research. I was also pleased to have Steve serve on my qualifying examination committee. I'm very fortunate to have been part of Meta-UW AI Mentorship program developed by Luke, with Hongyu as my mentor. The program forged collaborations with some of

the best researchers in speech processing through the Seamless Communication team at Meta. This enabled me to tackle grand problems in real-time speech and audio. I'm extremely grateful for Hongyu for being patient with a myriad of questions, ad-hoc whiteboard discussions and overall making sure I was never stuck for either technical or logistical reasons.

My PhD journey was also made much better by my fantastic lab-mates. I'd like to thank Malek Itani, Tuochao Chen, Maruchi Kim, Justin Chan, Anran Wang, Vidya Srinivas, Antonio Glenn, and Guilin Hu for creating such a vibrant and collaborative lab atmosphere. Our thought-provoking discussions were always incredibly valuable. I owe particular thanks to Malek Itani and Tuochao Chen, who partnered with me to co-lead several key projects; your collaboration was invaluable. I was constantly amazed by Malek's versatility and his deep knowledge spanning the entire stack, from ML model development right down to designing PCBs and working with OEMs – turning ideas into reality alongside you was a truly incredible experience. Likewise, I can only aspire to match Tuochao's impressive ability to juggle so many parallel threads of work so effectively. I'm glad to have collaborated with Maruchi and Antonio on the IRIS project. The concept, techniques and the final prototype were all very elegant, and it was a pleasure working with them. I also got to work with very talented, then undergraduate students, Eyoel Gebre and Lucas Lee through the IRIS project. I was also fortunate to have worked with Michael's students Scott Davidson, Tommy Jung, Max Ruttenberg, Dan Petrisko and Paul Gao during my master's.

I am grateful to the Allen School staff, Elise, Joe, and Chris, for their efforts in streamlining the PhD process. Special thanks to Chris for the enjoyable opportunity to participate in the intramural softball tournament. Furthermore, the Allen School's vibrant community significantly enriched my PhD experience. I was fortunate to become acquainted with Aditya Kusupati, Sebastian Santy, Chien-Yu Lin, Sudheesh Singanamalla, Priyal Suneja, Reshabh Sharma, Ameya Patil, Tapan Chugh, Anandghan Waghmare, Ishan Chatterjee, Vivek Jayaram, Pratyush Patel, Siddharth Iyer, Prashant Rangarajan, Gantavya Bhatt, Sahil Verma, and Vishwas Sathish.

My visiting role at Meta was incredibly valuable thanks to the fantastic team: Chao-Wei Huang, Tu Anh Nguyen, Bokai Yu, Ben Peloquin, Min-Jae Hwang, and Ilia Kulikov. The evaluation of SyncLLM, a key project in this thesis, really wouldn't have been the same without the collaboration with Ben and Bokai. I really appreciate Chao-Wei and Tu Anh for helping me get up to speed with the codebase and for all the

great brainstorming sessions. Big thanks to Min-Jae and Ilia for the lunch chats – they definitely made my time in the Seattle office much more enjoyable. I'm also so grateful for the support and encouragement from our managers, Paden Tomasello, Sravya Popuri, and Juan Pino. I especially remember their positive feedback when I first showed the SyncLLM demo internally, which really gave me the confidence to pursue those bigger ideas.

Finally, I am eternally grateful for the unwavering love and support of my parents, Divakar and Uma Devi Veluri, my brother Mithun, and my wife Akhila. My parents' commitment to my education, even amidst financial hardship, was foundational to this achievement. It's not an exaggeration to say my brother is happy and excited than me whenever I succeed. My wife, Akhila, stood by me throughout the most demanding periods of my PhD journey. She was my rock during those times and I feel incredibly lucky to have someone so positive by my side.

DEDICATION

To my parents — whose sacrifices and commitment to my education made this a reality.

Contents

List of Figures

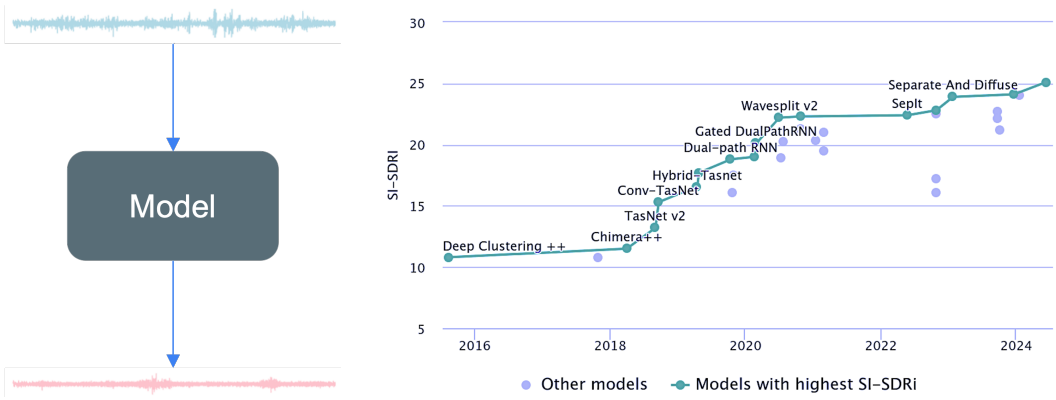
List of Tables

Chapter 1

Introduction

Like with many challenging problems in vision and text domains, deep learning has produced groundbreaking results in speech and audio processing. Speech recognition with has seen tremendous breakthroughs using neural networks with supervised Graves et al. [2006]; Hannun et al. [2014], self-supervised Baevski et al. [2020] and with weakly supervised Radford et al. [2023] methods. On the other hand, deep learning has made advances in text-to-speech to such an extent now that the generated voice is indistinguishable from natural human voice Le et al. [2023]; Wang et al. [2023a]. Neural networks have been shown to be very effective in sound classification and sound event detection Mesaros et al. [2018a, 2019] as well.

A special class of audio tasks involve *audio transformations*, where both the input and the output are audio waveforms. This includes tasks such as blind source separation Hershey et al. [2016]; Luo and Mesgarani [2019b]; Défossez et al. [2021], target source extraction Delcroix et al. [2022b]; Žmolíková et al. [2019] and speech-to-speech translation Barrault et al. [2023]; Jia et al. [2019]. Audio transformations can be categorized as causal or non-causal transformations depending on the temporal relationship between input and output waveforms. Causal audio transformations assume a common time frame of reference between input and output, and a result, at any point in time output signal can only depend on input signal until that point in time. Because, in a shared time frame of reference future audio samples do not exist, with respect to the current time step in the output signal. Examples of tasks involving non-causal transformations are audio summarization He et al. [1999], audio editing Wang et al. [2023c], offline transcription and translation Radford et al. [2023]. Whereas noise cancellation, streaming speech-to-speech translation and LLM-user



(a) Audio transformation

(b) Source separation performance

Figure 1.1: Audio transformations are audio processing tasks where both input and output are audio signals. A step change in performance as deep learning methods became popular.

voice interactions ar examples of causal audio transformations. Deep learning has achieved exceptional results in non-causal versions of these tasks Chen et al. [2023]; Barrault et al. [2023]. Figure 1.1 shows the performance improvement in speech separation over time with a metric equivalent to signal-to-noise ration, so higher is better. we can notice the jump in performance around the time deep learning started to become the method of choice.

Similar growth has not been seen in an important class of audio transformations that can be classified as real-time audio transformations. A key characteristic of real-time audio transformations is, input and output audio share the same exact time frame of reference. Unlike offline audio processing tasks where only the scale of time axis has to be preserved, here the shift in time axis has to be preserved as well and any shift in the time axis caused by our model would perceived as latency by the user. Shared time frame of reference has two interesting consequences in the relation between input and output audio signals: causality and equality in duration.

• Causality: Output at any timestep can only depend on input until that timestep as in a real-time setting, input beyond the current timestep does not exist.

• Equality in duration: Input and output audio signals would have exactly the same length because at any point in time, both input and output signals would have elapsed same amount of time in the shared time frame of reference.

Real-time audio transformations have some important applications. As shown in figure 1.2, in the very

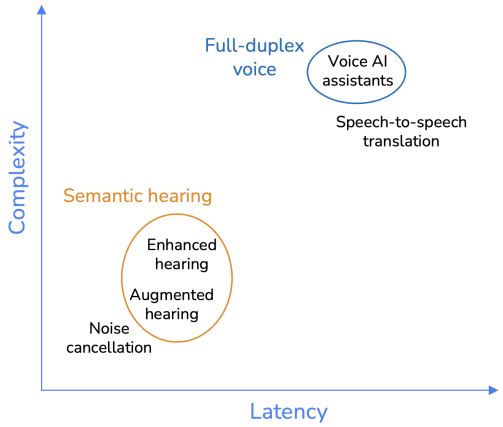


Figure 1.2: Applications of real-time transformations with different latency constraints and complexity. This report details methods for two classes of such real-time audio applications on the opposite ends of the spectrum.

low-latency regime, we have applications like noise-cancellation, hearing aids and enhanced hearing. And applications with relatively looser latency constraints, but significantly more complex are speech-to-speech translation and voice assistants. This report first presents several methods proposes for capabilities collectively referred to as *semantic hearing* Veluri et al. [2023a,c, 2024a]. Semantic hearing systems allow us program acoustic scenes in real-time, giving users the ability to chose what they want to hear in a live acoustic environment. Models developed for these tasks are intended to be deployed on wearable headsets and have the potential to both improve the quality of life for people with hearing disabilities as well as give us some hearing abilities we do not inherently possess. The following chapter discusses methods for developing *full-duplex voice* agents that require high-level understanding of human conversations Veluri et al. [2024b]. Finally, we investigate a key limitation in spoken language models trained with semantic speech representations that are speaker invariant Hsu et al. [2021]; Liu et al. [2023]. These representations treat all speech uniformly, including background speech interference. This makes it challenging for the dialogue model to distinguish the speaker it should listen to, from the interfering speaker. We show that in real-world situations where there is interfering speech, spoken dialogue models trained to understand acoustic representations are more robust to noisy speech.

Chapter 2

Semantic Hearing

Imagine you are in a park with pleasant sounds of birds chirping, but also cacophony of a nearby construction site, or you're in a busy restaurant trying to talk to the person before you, or you are walking in a busy street with mostly traffic noise, but also important sounds like car honks and emergency sirens. In all these scenarios, our brain is working hard to focus on the sounds or speech of interest while ignoring the interfering sounds Jafari et al. [2019]. Wouldn't it be nice to have a device extract the sounds of interest and reduce the cognitive load for us? This chapter discusses several techniques towards achieving the above set of capabilities, collectively referred to as *semantic hearing*. Semantic hearing systems are intended to be deployed on hearable devices that enables them to, in real-time, focus on, or ignore, specific sounds from real-world environments, while also preserving the spatial cues.

This task involves extracting an audio component from a given audio mixture in a streaming fashion. Several deep learning methods have been proposed for similar tasks Ochiai et al. [2020]; Luo and Mesgarani [2019b]; Libera et al. [2024], but they only do offline processing and do not demonstrate real-time streaming capabilities. In particular, the prior works need access to a large block ($\geq$ 1 s) of audio samples. In contrast, real-time streaming applications impose significant algorithmic and computational constraints, requiring networks to operate on small blocks ($\leq$ 10 ms) with a limited number of lookahead samples for each block. To address this, we propose the first real-time binaural neural network for target sound extraction Veluri et al. [2023a,c].

We then propose methods to extend the capabilities of above system to also provide the user with the

Figure 2.1: Semantic hearing applications. a) Users wearing binaural headsets can attend to speech while blocking out only the vacuum cleaner noise, b) block out street chatter and focus on the sounds of birds chirping, c) block out construction noise yet hear car honks, and d) a meditating user could use headsets to block out traffic noise outside yet hear alarm clock sounds.

ability to chose *which person* they want to hear, in a noisy environment. We refer to this capabilty as *target speech hearing*. But unlike with the target sound extraction, where everyone has a consistent set of sounds they might want to hear (door knocks, car honks etc.), each user might want to different set of people, and as result in is impractical to bake in target speaker's voice characteristics into the neural network weights. This create a unique user interface problem with no trivial way for the user let the system know who they want to hear. We propose a novel method of using headset wearer's gaze as cue for the target speech hearing system. Finally, we discuss how we could train target sound extraction and target speech hearing models to generalize to different acoustic environement and wearers.

We first start with an overview of related work, then describe our system requirements, and then present the network architecture we use for real-time binaural target sound extraction on smartphones. Next, we present our training methodology that generalizes our design to real-world use.

2.1 Background

Over the last decade, noise-canceling headsets and earbuds have undergone significant improvements, which now allow for more effective attenuation of *all* sounds in the environment. In fact, our experiments, where we play white noise to a human subject wearing a pair of Sony WH-1000XM4 headphones, show the impressive attenuation capabilities of these modern systems (Fig. 2.2). We identify this as an opportunity that provides us with an acoustic clean slate to introduce back target binaural sounds of interest from the environment. To the best of our knowledge, none of the prior work has explored semantic hearing capabilities for hearables. In the rest of this section, we describe related work in hearable systems, signal processing and machine learning for audio, and interaction tools.

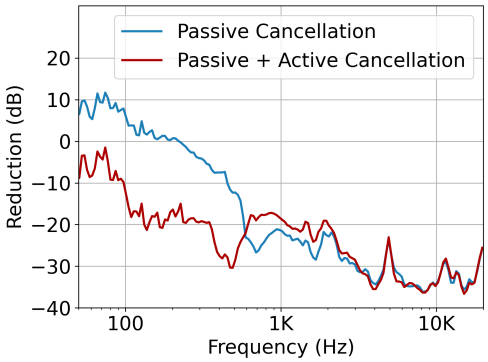


Figure 2.2: Noise reduction achieved with Sony WH-1000XM4 headphones – with and without active noise cancellation turned on – measured using an in-ear microphone inside the headphone cup. The reduction is measured relative to a microphone recording outside the ear cup. The spuriously large values at low frequencies (< 100 Hz) are due to the in-ear microphones picking up the wearer's blood pulse.

Active noise cancellation and acoustic transparency. Active noise cancellation is a well-studied problem where outward-facing microphones are used to capture sounds Shen et al. [2018]. An anti-noise signal is then transmitted to cancel *all* the external sounds and noise, which has more stringent delay requirements than semantic hearing. Traditional noise cancellation systems required bulky headsets. However in recent years lightweight in-ear earbud systems like the AirPods Pro can achieve reasonable noise-cancellation in many practical scenarios air [2023]. Semantic hearing leverages noise-cancelling earphones to cancel *all* sounds and then uses the mechanisms in this paper to program acoustic scenes in real-time.

The acoustic transparency mode for in-ear devices tries to imitate the sound response of an open-ear system by transmitting the appropriate signals into the ear canal Jin et al. [2022]. Like active noise cancellation, this is agnostic to the sound classes. Adaptive transparency on Apple airpods is designed to automatically reduce the amplitude of loud sounds ada [2023]. While related, this does not allow the user to pick and choose which sound classes to hear.

Speech systems. Prior systems have predominantly focused on improving the performance of speech-related tasks for in-ear devices (e.g., Airpods), telephony (e.g., Microsoft Teams), and voice assistants (e.g., Google Home). This includes speech enhancement Chatterjee et al. [2022]; McDonnell et al. [2023], target speech extraction Eskimez et al. [2022]; Giri et al. [2021], and speech separation Subakan et al. [2021]; Luo et al. [2022]. Oftentimes, these systems collectively regard all non-speech sounds just as noise. In contrast, semantic hearing requires understanding the semantics of various natural and artificial sounds in real-time, in the presence of interfering sounds, and determining which sounds to allow and which to block, based on

user input. Speech is one amongst many other sound classes in our system.

Neural networks for target sound extraction. Target sound extraction is the task of separating one or a limited number of target sounds from a mixture of sounds. Compared with speech systems, this is an underexplored problem in the audio machine learning community. However recent works have proposed neural networks that can achieve target sound extraction where clues about the target sound are provided either via audio Delcroix et al. [2022a]; Gfeller et al. [2021], images Gao and Grauman [2019]; Xu et al. [2019], text Kilgour et al. [2022]; Liu et al. [2022], onomatopoeic words Okamoto et al. [2022], or a one-hot vectors Ochiai et al. [2020]. All these models are designed for offline processing of audio clips, where the neural network has access to the entire audio file ($\geq$ 1 s) and hence cannot support our real-time hearable use-case.

The closest related work is our recent research on Waveformer Veluri et al. [2023b], which introduces a neural network architecture for target sound extraction. Waveformer was shown to run in real-time on a desktop computer. Our work differs from Veluri et al. [2023b] in two important dimensions. First, Waveformer is a single-channel model that operates on a single microphone. In contrast, our target use-case requires binaural processing across the two ears. Second, all prior work in this domain was evaluated on synthetic datasets and has not been demonstrated on hardware in real-world scenarios. In contrast, we present the first binaural target sound extraction system that can run in real-time on smartphones. We designed a training methodology that allows our system to generalize to unseen indoor and outdoor real-world environments.

Hearable applications. Recent work has used in-ear sensors for health applications Bui et al. [2021]; Chan et al. [2019, 2022] and activity tracking Ma et al. [2021]; Prakash et al. [2020]. Prior work has also explored various interaction modalities like ultrasound sensing Wang et al. [2022b] and on-face interaction Xu et al. [2020] for in-ear devices. The closest to our work is Clearbuds Chatterjee et al. [2022], which focuses on the task of enhancing the speech of the wearer using synchronized audio signals from two wireless earbuds. This prior work is focused on speech enhancement and is complementary to our system. Further, since the target application for Chatterjee et al. [2022] is telephony, it uses a 44.8 ms lookahead and has a latency of 109 ms.

Audio-based tools. Prior work has explored the use of sounds Yatani and Truong [2012]; Lu et al. [2009]; Mollyn et al. [2022]; Laput et al. [2018]; Tonami et al. [2022]; Jain et al. [2020b,a] to perform

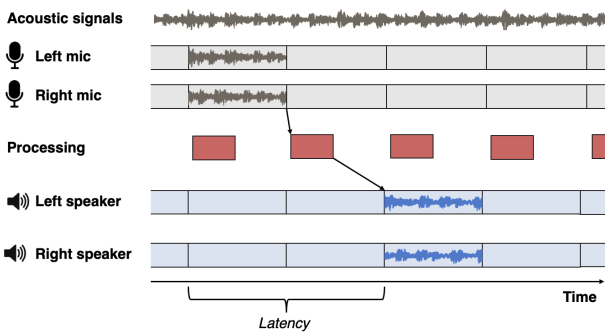


Figure 2.3: System requirements. Different components that contribute to latency in binaural target sound extraction.

activity recognition for wearables and smart home applications. These systems operate on around 1s audio chunks as the target use cases do not have the O(10 ms) latency requirements of in-ear audio applications. Prior work has also designed interaction tools for audio editing Rubin et al. [2013]; Pavel et al. [2020]. Our work is complementary in that it is focused on in-ear audio applications and semantic hearing that has more stringent latency requirements.

2.2 System Requirements

The goal of our design is to program the acoustic environment with imperceptible latency such that the target sound of interest is present but all other interfering sounds are suppressed. Given the stringent latency constraints, we cannot perform the necessary computation in the cloud but have to operate in real-time using computationally constrained devices like smartphones. Further, the target sounds generated by the model must originate from the same spatial directions as the real-world target sounds. Thus, our design must meet two key requirements: 1) real-time low-latency operation, and 2) binaural real-world generalization.

Real-time low-latency operation. Fig. 2.3 shows the different components that contribute to end-to-end latency in binaural acoustic processing systems. The first step is to feed the sound signals into two memory buffers of the binaural microphones. The acoustic data from the two microphones in each block is then fed into our neural network that outputs a block-length worth of binaural target sound data. This binaural output is then played back through the two speakers on the headset.

To ensure that the audio played through the headset is synced with the user's visual senses, we need this end-to-end latency to be less than 20-50 ms Stone and Moore [1999]; Gupta et al. [2020]; Wang et al.

[2022a]. To achieve this, we need to reduce the buffer duration, the look-ahead duration and the processing time. This is challenging for multiple reasons. 1) A small buffer duration of say 10 ms means that the algorithm has only an 10 ms block of current data to not only understand the semantics of the acoustic scene but also separate the target sound from other interfering sounds. While we can use the acoustic signals that arrived prior to the current block, many of our target sounds (e.g., door knocks) are not continuous. Reducing the buffer size even further to say 2 ms can be challenging from an operating system perspective since it can increase the number of system calls. 2) While a large lookahead can provide more context for the neural network to extract the target sounds, meeting our end-to-end latency requirement reduces the leeway we have in terms of the available lookahead to a few milliseconds. 3) Real-time operation requires processing each acoustic block within the duration of the block itself. This means that it should take less than 10 ms to process a 10 ms buffer Wang et al. [2022a]. This can be challenging since neural networks are not known for their lightweight computation. Further, since we cannot send the data to the cloud, the processing must be performed on-device on computationally-constrained devices like smartphones. In addition to all the above constraints, the operating systems also has I/O delays which for audio on iOS is on the order of 4 ms, depending on the buffer size ios [2023].

Binaural real-world generalization. In real life, the target sounds experience reverberations and multipath propagation due to reflections from walls and other objects in the environment. Further, the human head and torso reflect and obstruct sounds. As a result the target sound arrives with different amplitudes and delays at the two ears. The differences in the received sounds across the two ears provide spatial awareness to humans. Thus, it is critical in our design to preserve these differences and play the target sounds with different amplitudes and delays through the two speakers of the headset. This is challenging since the target and interfering sounds can be at different positions and experience different reverberations and reflections from the head-related transfer function. Further, the multipath effects and reverberations are difficult to predict in real-world environments, let alone the fact that the head-related transfer functions can change across wearers.

2.3 Real-time target sound extraction

Humans are exceptionally adept at attending their auditory focus to specific sounds even in a noisy environment Ochiai et al. [2020]. Recent works that aim to create a computational equivalent of this human capability formulate this problem as target sound extraction Ochiai et al. [2020]; Delcroix et al. [2021, 2022a]. The goal is to extract sound signals of interest from a mixture of various overlapping sounds, given clues which provide information about the target sound class such as embeddings of a one-hot label Ochiai et al. [2020], audio clips Delcroix et al. [2022a]; Gfeller et al. [2021], and images Gao and Grauman [2019]; Xu et al. [2019]. Streaming target sound extraction could enable real-time intelligent acoustic applications for headphones, hearing aids and telephony by filtering out undesired sounds from the environment (e.g., traffic) and presenting only sounds of interest to the user (e.g., sirens).

Recent works on target sound extraction have shown promising performance even for mixtures containing a large number of sound classes Ochiai et al. [2020]. However, none of these prior works demonstrate real-time streaming capabilities. In particular, the prior works for this task are based on non-streaming models and designed for offline processing, where the neural network has access to a large block (≥ 1 s) of audio samples Ochiai et al. [2020]. In contrast, real-time streaming applications impose significant algorithmic and computational constraints, requiring networks to operate on small blocks (≤ 10 ms) with a limited number of lookahead samples for each block. All these factors can significantly degrade the performance Luo and Mesgarani [2019a].

In this paper, we present the first deep learning method to perform target sound extraction in a streaming manner. Fig. 2.4 shows Waveformer, our encoder-decoder architecture where the encoder is a stack of dilated causal convolution (DCC) layers and the decoder is a transformer decoder Vaswani et al. [2017].[1] Our intuition is that much of the complexity in prior models comes with processing large receptive fields, especially at high sampling rates. For example, recent transformer based architectures proposed for speech separation Subakan et al. [2022]; Luo et al. [2022] implement chunk-based processing, where each chunk independently attends to all the chunks in the receptive field. Thus, to achieve a receptive field of length R, for each chunk, these models have an $\mathcal{O}(R)$ computational complexity. Instead, since DCC layers have a

[1] We call our network, Waveformer, since it uses an hybrid architecture with the causal convolution layers, common in WaveNet Oord et al. [2016] based architectures, as the encoder and a transformer as the decoder.

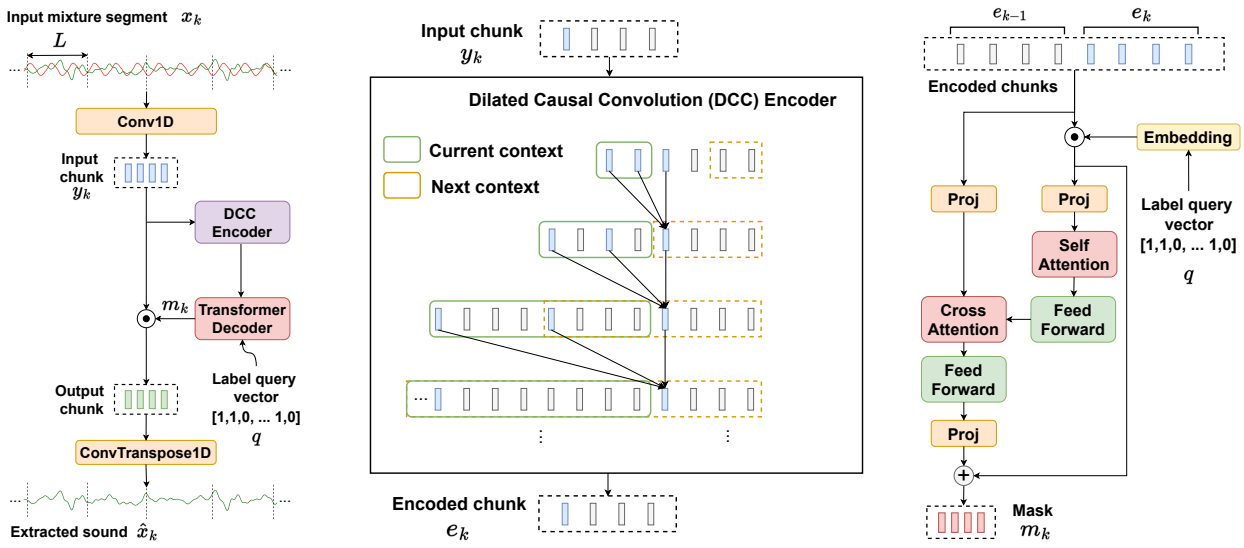
complexity of $\mathcal{O}(\log R)$ for achieving the same amount of receptive field (§2.3.3), we use a stack of DCC layers as the encoder that processes the receptive field. We then use the decoder layer of the transformer architecture Vaswani et al. [2017] as our model's decoder. We leverage the *memory–target* paradigm of the transformer decoder, where the query-conditioned encoder's output is considered as the transformer decoder's 'target' and the unconditioned encoder's output is considered as the transformer decoder's 'memory'. Using self-attention on the 'target' followed by cross-attention between the 'target' and 'memory,' the decoder generates the mask for extracting the specified target sound to produce the output signal.

To evaluate our network architecture, we implement a causal version of Conv-TasNet and a streaming version of ReSepformer Subakan et al. [2022] for the task of streaming target sound extraction. Evaluations show that our hybrid network architecture achieves state-of-the-art performance for this task. Further, the smallest and largest versions of our model have real-time factors (RTFs) of 0.66 and 0.94, respectively, on a consumer-grade CPU, demonstrating the real-time target sound extraction capability. This is shown to be faster than the two representative prior models while outperforming them in terms of the output signal quality.

2.3.1 Related work

Universal sound separation. The task here is to decompose a mixture of arbitrary sound types into their component sounds, regardless of the number of sounds in the mixture Kavalerov et al. [2019]. This becomes increasingly challenging as the number of possible sound types in the mixture increases. Several networks have been proposed for this task including convolutional long short-term memory networks Kavalerov et al. [2019], time-dilated convolution networks Kavalerov et al. [2019] based on Conv-TasNet Luo and Mesgarani [2019a], and transformer networks Zadeh et al. [2019]. Prior work also proposed the use of embeddings learnt by a sound classifier trained on a large sound ontology Gemmeke et al. [2017b] for conditioning a separation network.

Target sound extraction. This aims at separating one or a limited number of sounds of interest from a mixture by conditioning a source-separation model with query information or clues about the target sound. This approach can circumvent the challenge of universal sound separation struggling to deal with a mixture of a large number of sounds. The clues may be provided as an embedding of an audio clip Delcroix



(a) End-to-end architecture **(b)** DCC Encoder **(c)** Transformer Decoder

Figure 2.4: Waveformer architecture. Streaming inference demonstrated using an example input mixture segment of length $4L$ samples, corresponding to a chunk length $K = 4$. The query is a one-hot or a multi-hot label encoding. The Dilated Causal Convolution (DCC) encoder encodes the input chunk y_k using the context computed from the receptive field. The transformer decoder computes the target mask by attending to current and previous encoded chunks.

et al. [2022a]; Gfeller et al. [2021], an image Gao and Grauman [2019]; Xu et al. [2019], natural language text Kilgour et al. [2022]; Liu et al. [2022], onomatopoeic words Okamoto et al. [2022], or a one-hot sound label vector Ochiai et al. [2020]. The prior works Kong et al. [2020]; Chen et al. [2022] have also evaluated the use of a sound event detector to detect the time when the target sound occurs in a mixture. Although, these works are often motivated for practical usage Liu et al. [2022]; Ochiai et al. [2020], none of them use streaming models. In contrast, we design the first streaming network for target sound extraction using attention.

Speech-specific networks. Prior work has also focused on speech enhancement Wisdom et al. [2019]; Eskimez et al. [2022]; Chatterjee et al. [2022], speech separation Luo and Mesgarani [2019a]; Luo et al. [2020]; Huang et al. [2014]; Hershey et al. [2016]; Isik et al. [2016]; Yu et al. [2017]; Wang et al. [2018] and speech selection using clues provided to the network Tzinis et al. [2022]; Ephrat et al. [2018]; Wang et al. [2022a]. For speech enhancement, neural networks have been proposed Eskimez et al. [2022] to realize real-time operation. Recently, efficient transformer based architectures have been proposed for speech recognition

and separation tasks Subakan et al. [2021]; Luo et al. [2022]; Gulati et al. [2020]. These method either use standard transformer blocks Subakan et al. [2022] or convolution augmented transformer blocks Luo et al. [2022]; Gulati et al. [2020]. In contrast, we use DCC layers to process the receptive field and a transformer decoder for generating the target mask for sound extraction. Recent work on ReSepformer Subakan et al. [2022] proposed a causal mode for their transformer method, which we use for our baseline comparisons.

2.3.2 Waveformer Architecture

We process individual audio chunks of duration τ seconds. For streaming, we need to operate at chunk level: an output chunk can depend on the current and past chunks. Thus, streaming models have an intrinsic latency equal to the duration of a single chunk. In real-time practical systems, it is desirable that this latency is on the order of 10 ms Sunohara et al. [2017]. Fig. 2.4 shows our proposed time-domain model architecture, which employs an encoder-decoder based mask generation network to generate an element-wise multiplicative mask in the latent space. Let $x_k \in R^S$ denote the current input audio chunk, where $S = \tau F_s$ is the number of audio samples included in the current chunk with a sampling rate of F_s. In the first step, a 1D-convolution layer with stride L and kernel size $3L$ is applied to the input audio chunk x_k to obtain the latent space representation, $y_k \in R^{E \times K}$, where E is the latent space feature dimensions and $K = \frac{S}{L}$ is the feature sequence length in the latent space. Setting the kernel size to $3L$ and stride to L requires an overlap of L samples with the previous and the future chunk, resulting in a lookahead of L samples. In our experiments, we set $L = 32$ samples at 44.1 kHz. This results in a lookahead of around 0.73 ms, which is negligible. Given a one-hot or multi-hot query vector $q \in \{0, 1\}^{N_c}$, where N_c is the total number of classes, streaming target sound extraction is achieved by computing feature masks $m_k \in R^{E \times K}$. With the mask generation network and element-wise multiplication denoted as $\mathcal{M}$ and $\odot$, respectively, the target sound signal, $\hat{x}_k \in R^S$, is computed as:

$$
\begin{aligned}
y_k &= \text{Conv1d}(x_k), \quad m_k = \mathcal{M}(y_k \mid y_{k-1}, .., y_2, y_1, q) \\
\hat{x}_k &= \text{ConvTranspose1d}(y_k \odot m_k).
\end{aligned}
$$

2.3.3 Dilated causal convolution encoder

Our encoder is a stack of dilated causal convolution (DCC) layers Oord et al. [2016], and the decoder is a transformer network Vaswani et al. [2017]. The motivation for such an architecture is that the encoder computes a *contextful* representation of the input chunk, considering the previous chunks up to a certain receptive field, and the decoder conditions the encoder output with the query vector to estimate the target mask. While recent transformer models for speech separation Subakan et al. [2021, 2022] have demonstrated performance gains over convolution based methods Luo and Mesgarani [2019a], the latter have generally been more computationally efficient.

We attribute this efficiency gap to the difference in the way existing transformer models process the receptive field compared to convolution based architectures. To achieve a receptive field of length R, given the chunk based processing in existing transformer architectures, each chunk individually attends to all previous chunks in the receptive field resulting in $\mathcal{O}(R)$ complexity. In contrast, convolution based models Oord et al. [2016]; Luo and Mesgarani [2019a] using a stack of M DCC layers with kernel size P and exponentially scaling dilation factors $\{2^0, 2^1, 2^2, ..., 2^{M-1}\}$ have a receptive field of $(P-1) \cdot (2^M - 1)$. Its complexity is $\mathcal{O}(PM)$. With a small kernel size P, the computational complexity of the stacked DCC layers is $\mathcal{O}(P \cdot \log\,(1 + \frac{R}{P-1})) \sim \mathcal{O}(\log R)$ for it to have a receptive field of length R.

We use 10 DCC layers with a kernel size of 3 and dilation factors $\{2^0, 2^1, 2^2, ..., 2^{10-1}\}$ in our encoder, resulting in a receptive field of $(3-1) \cdot (2^{10} - 1) = 2046$ samples in the latent space. With the initial input convolution stride L set to 0.73 ms, our encoder's receptive field is $\approx 1.5s$. Fig. 2.4 (b) shows an encoding of an input chunk of length 4. For chunk based streaming inference, the encoder maintains a context buffer for each DCC layer. This context is initially computed from the $1s$ receptive field, and then updated dynamically after encoding each subsequent chunk. For encoding a chunk, each DCC layer is fed with the output chunk of the previous layer, left padded with the context of length twice the layer's dilation. After encoding a chunk, the context is updated with the rightmost elements of the padded input for it to be used in encoding the next chunk. For each input chunk y_k, the DCC encoder computes an encoded representation, $e_k \in R^{E \times K}$.

Table 2.1: Performance for the single-target extraction task. In our model, E and D correspond to encoder and decoder dimensionalities, respectively. RTF is the real-time factor for a consumer CPU.

Model	Model size	RTF	SI-SNRi
Conv-TasNet	4.57M	1.34	6.14
ReSepformer	13.24M	1.60	7.26
Ours ($E = 256;\ D = 128$)	1.10M	0.66	9.02
Ours ($E = 256;\ D = 256$)	1.69M	0.75	9.40
Ours ($E = 512;\ D = 128$)	3.29M	0.88	9.26
Ours ($E = 512;\ D = 256$)	3.88M	0.94	**9.43**

2.3.4 Query-conditioned transformer decoder

To get the mask, the encoded representation computed above must be conditioned with the query, q. To this end, we first compute an embedding, $l \in R^{E\times 1}$, corresponding to q. This is achieved by using an embedding layer comprising three 512-dimensional feed-forward sub-layers with an N_C-dimensional input and an E-dimensional output. Our transformer decoder conditions the encoded chunk e_k with the query embedding l and derives the mask as follows.

Fig. 2.4 (c) shows our decoder architecture. First, we perform multiplicative query integration Ochiai et al. [2020]; Delcroix et al. [2022a] to compute the conditioned representation: ${e_k}' = e_k \odot l$. Since transformers are more computationally expensive with higher dimensionality, we first project the encoded representations, ${e_k}'$ and e_k, to the decoder dimensions $D \leq E$ with 1×1 convolution. This results in projected encoded representations, $pe_k, {pe_k}' \in R^{D\times K}$. The decoded representations are then computed by passing $pe_k, {pe_k}'$ to the transformer decoder layer's self-attention and cross-attention blocks, respectively, to obtain target mask $pm_k \in R^{D\times K}$ in the projected decoder space. It is then projected back to the encoder dimensions with another 1×1 convolution layer to obtain ${m_k}' \in R^{E\times K}$. Since the bottleneck caused by the projection layers might affect the flow of gradients, as depicted in the diagram, we use a skip connection immediately after the multiplicative query integration to the output of the projection layer to compute the final mask: $m_k = {m_k}' + {e_k}'$.

Within the decoder, we use the chunk-based streaming attention scheme proposed in Chen et al. [2021]. As shown in Fig. 2.4(c), for decoding the current chunk, e_k, the transformer decoder only attends to the samples in the current chunk, e_k, and one previous chunk, e_{k-1}. This ensures that the input length to the transformer decoder is fixed at $2K$ (current chunk + one previous chunk) and prevents the inference time

from growing as the input audio length increases.

2.3.5 Experiments and results

Dataset. We use a synthetic sound mixture dataset created from the FSD Kaggle 2018 dataset Fonseca et al. [2018]. FSD Kaggle 2018 is a set of sound event and class label pairs, with 41 different sound classes, which are a subset of the Audioset ontology Gemmeke et al. [2017b]. Our synthetic dataset consists of 50k training samples, 5k validation samples and 10k test samples. Sound mixtures are created using the Scaper toolkit Salamon et al. [2017], each with 3-5 foreground sounds randomly sampled from the FSD Kaggle 2018 dataset, and a background sound randomly sampled from the TAU Urban Acoustic Scenes 2019 dataset Mesaros et al. [2018b]. Foreground sound classes are randomly sampled without replacement so that each sample has 3-5 unique classes. We construct the sound mixtures by sampling 3-5s crops from each foreground sound and then pasting them on a 6s background sound. The SNRs of the foreground sounds are randomly chosen between 15 and 25 dB, relative to the background sound. Our training and validation data are sampled from the development splits of FSD Kaggle 2018 and TAU Urban Acoustic Scenes 2019, while our test samples are from the test splits. From each mixture sample, 3 foreground sounds are randomly selected to be the target sounds. Only one of those 3 foreground sounds is used for the single-target extraction task, while others are used for the multi-target extraction task. The choices of the target foreground sounds are fixed after generating the validation and test sets, to ensure evaluations are reproducible. During training, however, the choices of the target foreground sounds in the training set are randomized. Since we mainly consider human listening applications for streaming target sound extraction, we run our experiments at a 44.1 kHz sampling rate to cover the full audible range.

Evaluation setup. Prior works Ochiai et al. [2020]; Delcroix et al. [2022a] show that Conv-TasNet, originally proposed for speech separation, can also be used for target sound extraction. Further, ReSepformer proposes an efficient transformer architecture for speech separation that allows a streaming inference. Here, we compare the performance of our architecture with the causal or streaming implementations of Conv-TasNet and ReSepformer as described in the original papers Luo and Mesgarani [2019a]; Subakan et al. [2022] for the target sound extraction task.

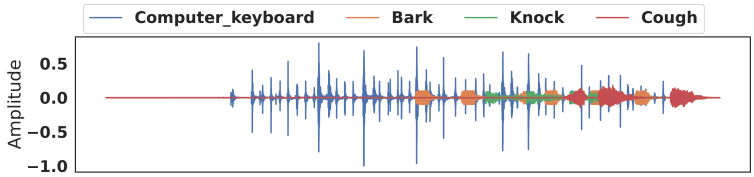
For all the models, we set the stride of the initial convolution, L, to 32, which is about 0.73 ms at 44.1

kHz. We train multiple configurations of our model with different encoder and decoder dimensions. We fix the number of DCC layers to 10, number of transformer layers to 1 and chunk length, K, to 13. This chunk length corresponds to 416 samples in the time domain, or a chunk duration of 9.43 ms. For Conv-TasNet, we follow the configuration used in Ochiai et al. [2020] except for the number of repeats, which we set to 2. This ensures that the runtime of the Conv-TasNet baseline is not too large compared with that of our model's largest configuration. For the ReSepformer baseline, we set the model dimensionality to 512, number of blocks to 2, number of transformer layers to 2 and chunk size to 13 (9.43 ms). We perform label integration after the first transformer block, as we found that to be perform better than integrating it at the beginning.

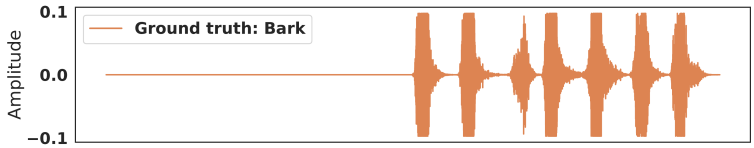
Loss function and training hyper-parameters. We use a linear combination of 90% signal-to-noise-ratio (SNR) and 10% scale-invariant-signal-to-noise-ratio (SI-SNR) Roux et al. [2018] as the loss function for training. While a scale dependent loss, such as SNR loss, is essential for preserving the original amplitude of the desired sounds in the mixture, we observed that adding a fraction of the SI-SNR helps stabilize the training process especially for causal models. We set the initial learning rate to 5e-4 for our models and Conv-TasNet, and to 1.5e-4 for the ReSepformer. We use the `ReduceLROnPlateau` learning rate scheduler to scale the learning rate by 0.1, if there is no improvement in the validation SI-SNR for more than 5 epochs. We use SI-SNR improvement (SI-SNRi) as the validation and test metric. We train the models for 100 epochs. We pick the weights after the epoch that resulted in the best validation SI-SNRi.

Results. We separately train the models for single-target and multi-target extraction tasks, and evaluate them on our testset. For multi-target evaluation, we train our model as well as baselines to make predictions with multi-hot query vectors, as opposed to one-hot queries used in the single-target evaluation. During the multi-target training, 1-3 foreground sounds are randomly selected as target sounds. This training method using arbitrary number of target sources helps the model learn multi-target embeddings. The same model configurations are used for both the single-target and multi-target experiments. The Conv-TasNet and ReSepfromer baselines are also trained in the same way for the multi-target extraction task.

We also evaluate the real-time factors (RTFs) of the models on an Intel Core i5 CPU using a single thread. RTF is computed by measuring the runtime consumed by the models to process a 416 sample audio chunk (9.43 ms at 44.1 kHz), and dividing that with the chunk duration, 9.43 ms. For the RTF measurement, we

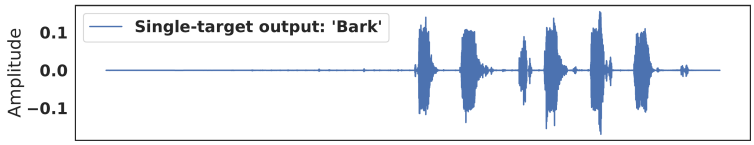


(a) Input mixture

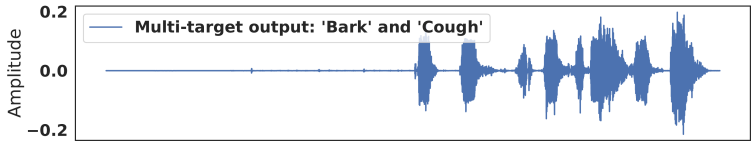


(b) Ground-truth for label 'Bark'

(c) Ground-truth for label 'Cough'



(d) Output of the single-target extraction of 'Bark'



(e) Output of the multi-target extraction of 'Bark' and 'Cough'

Figure 2.5: Visualization of time-domain waveforms of single-target and multi-target extraction (x-axis represents time).

include the padding for dilated convolution layers in our model's DCC encoder and Conv-TasNet's Temporal Convolution Network (TCN) blocks, accounting for the entire receptive field. In case of the ReSepformer, using a single chunk for RTF measurement excludes the overhead caused by causal attention masking in its inter-attention blocks. Consequently, the RTF value reported for ReSepformer is a lower bound of what is practically achievable.

Table 2.1 compares our models with different configurations with the baselines in terms of both the efficiency and the performance. We show that our approach results in 2.2-3.3 dB SI-SNRi improvement compared with the baselines, while being 1.5-2x more computationally efficient with 1.2-4x fewer parameters. Table 2.2 compares the performance of our models with the baselines for the multiple target extraction

Table 2.2: SI-SNRi comparison in the multi-target extraction task.

Model	# selected classes 1	2	3	Mean
Conv-TasNet	7.20	3.63	0.19	3.67
ReSepformer	7.42	3.56	0.33	3.77
Ours ($E = 256;\ D = 128$)	9.06	4.78	1.51	5.11
Ours ($E = 256;\ D = 256$)	9.12	4.76	1.31	5.06
Ours ($E = 512;\ D = 128$)	9.39	4.92	1.39	5.23
Ours ($E = 512;\ D = 256$)	9.29	4.92	1.35	5.19

task. It shows that our method outperforms the baselines by 1.2-1.4 dB for the 2-target case, and 1-1.2 dB for the 3-target case. As with prior work Ochiai et al. [2020], the SI-SNR improvements are lower in the 3-target selection task since there is greater similarity between the input mixture and the target signal, compared to the single-target case, resulting in a larger input SI-SNR.

In Fig. 2.5, we qualitatively show an example single-target extraction and multi-target extraction from a 4-class input mixture, using our multi-target extraction model. Fig. 2.5a shows the input mixture waveform, and Figs. 2.5b and 2.5c show the isolated ground-truth sounds. We provide the input mixture to our multi-target model, with a single target query followed by a two target query. Figs. 2.5d and 2.5e are the output waveforms obtained when the single target and the two targets are queried, respectively. The waveforms show that the model successfully recognizes the queried events and extracts the relevant sounds. It can also be observed that our model preserves the original amplitudes of the sounds in the input mixture well.

The use of higher dimensionality such as 512, in DCC layers is shown to perform better Luo and Mesgarani [2019a]. Using the 512 dimensions in the transformer decoder, however, can make it computationally intensive. So, it is better to run the DCC encoder at higher number of dimension than that of the transformer decoder. Since we use projection layers to convert between the encoder and decoder dimensions, to mitigate the effects the projection layers have on gradients, we include a skip connection in the decoder. Our ablation study shows that the skip connection improves the SI-SNRi from 9.06 to 9.43.

2.3.6 Summary

We demonstrate the first deep learning method for real-time and streaming target sound extraction. Future work includes the use of more constrained computing platforms, larger datasets with more classes, and multiple microphones. Our Waveformer architecture may be applicable to other acoustic applications like

source separation and directional hearing, which deserves further exploration.

2.4 Real-time binaural target sound extraction

This section first discusses the high-level framework for our binaural target sound extraction neural network, followed by the causal and streaming adaptation of this network.

High-level framework

Consider $s \in R^{2\times T}$ to be the input binaural signal provided to the target sound extraction network. Since time-domain models also have been shown to be able to learn representations analogous to STFT features Luo and Mesgarani [2019b], our network operates on time-domain binaural signals. As shown in Fig. 2.6a, the signal is first mapped to a representation in a latent space, $x \in R^{D\times(T/L)}$, by using a 1D convolution layer with a kernel size $\geq L$ and a stride equal to L. D and L are tuneable hyperparameters of the model. D is the dimensionality of the model, having a significant effect on the parameter count, and consequently the computational and memory complexities. L determines the duration of the smallest audio chunk that can be processed with the model. The latent space representation x, is then passed to a mask generator, $\mathcal{M}$, which estimates an element-wise mask m as:

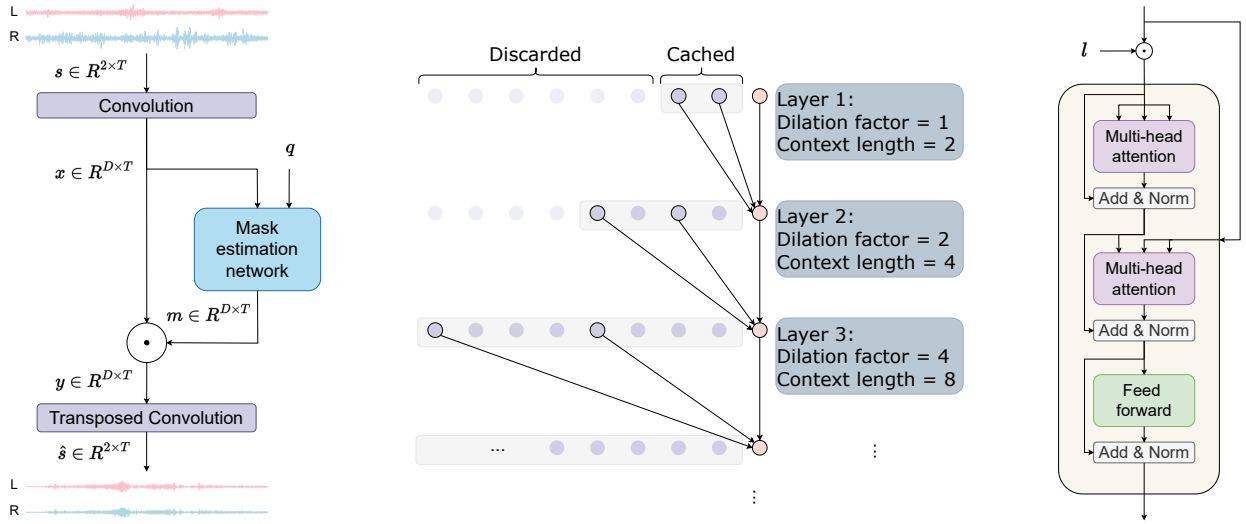
$$m = \mathcal{M}(x, q) \mid m \in R^{D\times(T/L)};\ q \in \{0,1\}^{N_c}, \tag{2.1}$$

where N_c is the total number of sound classes the model is trained for. The representation corresponding to the target sound is obtained by element-wise multiplication of the input representation, x, and the mask, m, as follows:

$$y = x \odot m \mid y \in R^{D\times(T/L)}. \tag{2.2}$$

The output audio signal $\hat{s} \in R^{2\times T}$ is then obtained by applying a 1D transposed convolution on y, with a stride of L.

Our design jointly processes the two channels for computational efficiency. In our experiments, we show that our simpler framework performs competitively with the prior parallel processing frameworks in terms of target sound extraction accuracy, even with a 50% lower runtime cost.



(a) Binaural extraction framework

(b) Encoder

(c) Decoder

Figure 2.6: Binaural target sound extraction network architecture. a) Our high-level binaural extraction framework. Mask estimation network is an encoder-decoder architecture operating on latent space representation of binaural signals to extract mask for target sound based on the query vector q. b) and c) show the encoder and decoder architectures used in the mask estimation network. The encoder processes the previous input context and does not consider the label embedding. Decoder first conditions the encoded representation with the label embedding, l, and then generates the mask corresponding to the target sound using the conditioned representation.

Streaming inference and causality

For real-time on-device operation, the model must output the audio corresponding to the target sound as soon as the input audio is received, i.e., with an end-to-end latency to be less than 20-50 ms Denk et al. [2020]. Since the audio is fed to the model from the device buffers, the buffer size determines the duration of the audio chunk the model receives at each time step. Assuming the buffer size to be divisible by the stride size L, the audio chunk size can be represented as the number of strides, K. That is, the buffer size of an audio chunk of size K is equal to KL samples. Such a real-time setup means that the model only has access to the current and previous chunks, but not future chunks. *This requires the model to be causal with the time resolution of the buffer size, i.e., KL audio samples.* As a result, in the high-level framework described above, the input convolution, the mask estimation block, the element-wise multiplication, and the output transposed convolution must operate on one audio chunk at each time step.

The binaural target sound extraction framework described above can be adapted to chunk-wise streaming

inference as follows. Consider the input audio signal corresponding to the kth chunk to be $s_k \in R^{2\times KL}$. The input 1D convolution maps this audio chunk to its latent space representation, $x_k \in R^{D\times K}$. The mask estimation block is then used to estimate the mask corresponding to the target sound, based on the current chunk, as well as a finite number of the previous chunks:

$$m_k = \mathcal{M}(x_k, q, x_{k-1}, x_{k-2}, ...) \mid m_k \in R^{D\times K} \tag{2.3}$$

The previous chunks act as the audio context for the neural network, referred to as the receptive field of the model. A receptive field of 1-1.5s is shown to result in good performance Luo and Mesgarani [2019b]. The output representation of the current chunk corresponding to the target sound, $y_k \in R^{D\times K}$ can then be obtained as:

$$y_k = x_k \odot m_k \tag{2.4}$$

The resulting output representation is then converted to the output signal $\hat{s}_k \in R^{2\times KL}$ by applying the 1D transposed convolution.

Mask estimation network

Several architectures have been proposed in the literature for mask estimation such as Conv-TasNet Luo and Mesgarani [2019b], SepFormer Subakan et al. [2021], ReSepFormer Libera et al. [2024], and Waveformer Veluri et al. [2023a]. Waveformer is an recently proposed efficient streaming architecture implementing chunk-based processing, which makes it suitable for our task. In this work, we use a modified version of Waveformer to further increase efficiency without any loss in performance. The mask estimation network is an encoder-decoder neural network architecture, where the encoder is purely convolution-based and the decoder is a transformer decoder.

Encoder. Mask estimation in Eq. 2.3, involves processing many previous chunks in addition to the current chunk to obtain the mask corresponding to the current chunk. Repeated processing of the entire receptive field for each iteration could become intractable for a real-time on-device application. To mitigate this inefficiency, while achieving a large receptive field, our mask estimation network implements a Wavenet van den Oord et al. [2016] style dilated causal convolutions for processing the input and previous chunks. In

this work, for efficient on-device inference, we implemented the dynamic programming algorithm proposed in Fast Wavenet Paine et al. [2016]. As shown in Fig. 2.6b, higher efficiency is achieved by reusing the intermediate results computed in the previous iterations. The encoder function $\mathcal{E}$ processes the input chunk x_k and an encoder context ξ_k to generate the encoded representation of the input chunk:

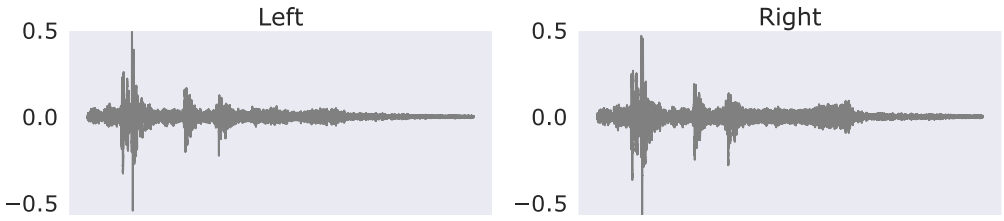
$$e_k, \xi_{k+1} = \mathcal{E}(x_k, \xi_k) \mid e_k \in R^{D \times K} \tag{2.5}$$

The size of the context ξ_k depends on the hyperparameters of the encoder. In our implementation, the encoder is comprised of a stack of 10 dilated causal convolution layers. The kernel size of all layers is equal to 3, and the dilation factor is progressively doubled after each layer starting with 1, resulting in dilation factors $\{2^0, 2^1, ..., 2^9\}$. Since the kernel size is equal to 3, the context needed for each dilated convolution layer is twice the layer's dilation factor. As long as this context is saved after each iteration, and padded with the input chunk in the next iteration, the intermediate results corresponding to the previous chunks do not have to be recomputed. Thus the size of the context ξ_k is equal to $2 \times \sum_{i=0}^{9} 2^i = 2046$.

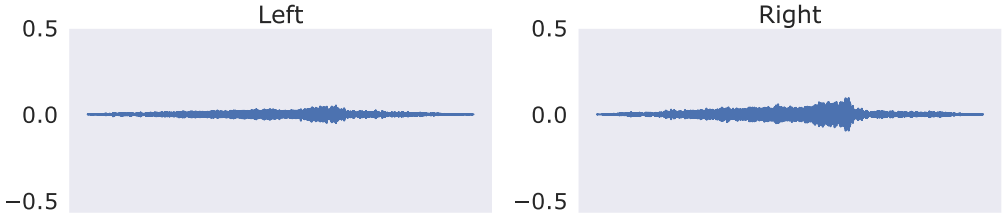
Decoder. The query vector q is first embedded into the embedding space using a linear layer to generate a label embedding $l \in R^D$. The mask corresponding to the target sound m_k is estimated using a transformer decoder layer Vaswani et al. [2023], represented here as $\mathcal{H}$. The encoded representation is first conditioned with the label embedding l by an element-wise multiplication. The encoded representation and the conditioned encoded representation are first concatenated in the time dimension, with those from the previous time step, before processing with the transformer decoder layer $\mathcal{H}$. The encoded representation from the previous time step, e_{k-1}, acts as the decoder context. The mask estimation can be written as:

$$m_k = \mathcal{H}(\{l \cdot e_{k-1}, l \cdot e_k\}, \{e_{k-1}, e_k\},) \tag{2.6}$$

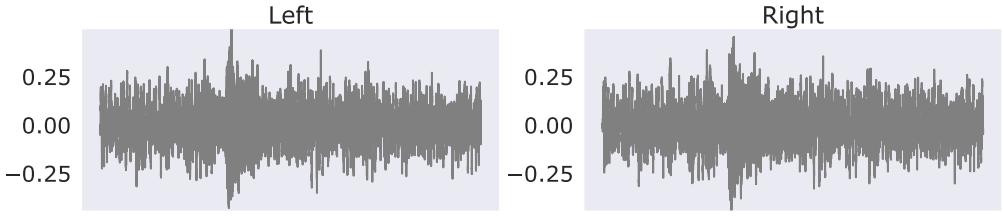
where $\{\}$ represents concatenation in the time dimension. As shown in Fig. 2.6c, the transformer decoder $\mathcal{H}$ first computes the self-attention result of the conditioned encoded representation $\{l \cdot e_{k-1}, l \cdot e_k\}$ using the first multi-head attention block, followed by cross-attention between the self-attention result and the unconditioned encoded representation $\{e_{k-1}, e_k\}$ using the second multi-head attention block. A feed-forward block along with residual connection generates the final mask corresponding to the target sound.



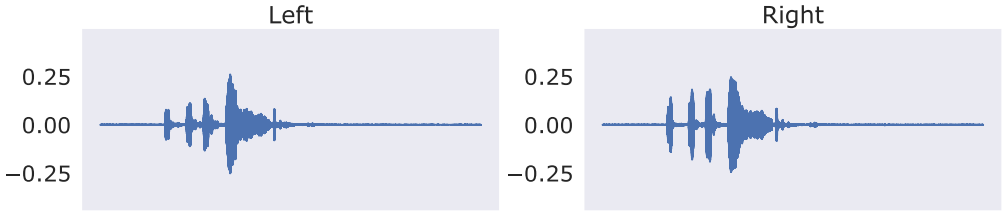
(a) Binaural input of emergency siren, chatter, and traffic noise.



(b) Binaural output with siren extracted.



(c) Binaural input of birds chirping, chatter, and street noise.



(d) Binaural output with birds chirping extracted.

Figure 2.7: Real-world binaural input and output recordings obtained with our semantic hearing system.

2.4.1 Results

We first describe our setup for real-world evaluations and then present our binaural network benchmarks.

Hardware prototype. Our hardware setup includes a pair of SonicPresence SP15C binaural microphones that are wired to capture high-quality recordings. We use an iPhone 12 to process the recorded data and output the audio through noise-canceling headphones like JBL Live 650BTNC and the NUBWO gaming headsets. We use a lightning-to-aux adapter to connect the headphones to the iPhone over a wire. We also use a USB hub to connect both the microphones and the headphones to the smartphone.

Participants. We recruiting 9 individuals (3 female, 6 male) across our in-the-wild and spatial cues evalua-

Figure 2.8: A participant in our in-the-wild evaluation where the target sound was birds chirping in the presence of urban environment noises. The participants could move their head freely and the target sound source could also be mobile.

tions. We also invited 22 participants (6 female, 16 male) for our online hearing study.

2.4.2 In-the-wild evaluation

To evaluate the proposed system in real-life scenarios, we conduct in-the-wild experiments to assess the effectiveness of our system.

In-the-wild scenarios. 5 individuals (3 female and 2 male) wore our hardware and collect sounds in the real world. These experiments were conducted in typical application settings: offices, living rooms, streets, rooftops, parks, and restrooms. Since some of the sound classes were relatively less common, our in-the-wild experiments had a subset of classes which most commonly appeared in our recordings: alarm clock, car horn, door knock, speech, computer typing, hammer, birds chirping, and music. The position and movement of the sound sources were uncontrolled and reflective of real-world scenarios, where the sound sources could be mobile. Furthermore, in all experiments, participants had complete freedom to move their heads, causing the sound source positions relative to the microphones to vary over time (Fig. 2.8). Thus, our in-the-wild evaluation captured both mobile wearers as well as mobile sound sources that naturally occured in real-world scenarios (e.g., cars moving or birds that fly).

Evaluation procedure. Unlike with our simulated training data, we do not have clean, sample-aligned ground truth signals to objectively compare the binaural outputs of our system with. Hence, we conduct a listening study to compute a mean opinion score (MOS) regarding the sound extraction accuracy. This metric is crucial to evaluate the perceptual quality of our algorithm for end-users, although it has often been

omitted in prior non-speech sound extraction research. We invited 22 participants (6 female, 16 male, mean age 34.6) to the online listening study. The study consists of 16 sections. In each section, the participants evaluated the quality of 3 or 4 5.0-8.5 second audio samples. The audio samples played at each section were in-the-wild recordings processed in the following three ways for the same target label: (1) the original recording, (2) the output of our 128-dimensional binaural network, (3) the output of our 256-dimensional binaural network. For the subset of the evaluations that involved speech as the target sound, we also included an additional fourth audio sample that was obtained by extracting of the interfering class (e.g., door knocks) and then subtracting it from the input recording to estimate the target speech.

We conducted a pre-screening process to ensure that the participants used suitable binaural headsets. This involved playing two white noise samples, one exclusively from the left channel and one exclusively from the right channel. The participants were instructed to confirm that they heard the sounds only from the correct channels. 11 of our participants used headphones, and 11 used earbuds during our online user study.

We measured the sound extraction quality based on both interference suppression and overall mean opinion score (MOS), as they are often included in speech enhancement quality assessment:

- **Noise suppression**: *How INTRUSIVE/NOTICEABLE were the BACKGROUND sounds? 1 - Very intrusive, 2 - Somewhat intrusive, 3 - Noticeable, but not intrusive, 4 - Slightly noticeable, 5 - Not noticeable*
- **Overall MOS**: *If the goal is to focus on the <TARGET> sounds, how was your OVERALL experience? 1 - Bad, 2 - Poor, 3 - Fair, 4 - Good, 5 - Excellent*

Results. In Fig. 2.9, we present the results of the user evaluations for the interference sound suppression and overall quality improvement of our system for different target sound labels. The results demonstrate the system's capability to significantly reduce unwanted background sounds, as indicated by an increase in the overall noise suppression score from 2.01 (corresponding to *2 - Somewhat intrusive*) to 3.61 (between *3 - Noticeable, but not intrusive* and *4 - Slightly noticeable*) with the 128-dimensional model, and to 3.84 (slightly worse than *4 - Slightly noticeable*) with the 256-dimensional model. We also observed a similar trend in the overall MOS improvement, with an improvement from 2.63 for the input signal to 3.54 and 3.80 after processing with the 128-dimensional and 256-dimensional models, respectively. Figs. 2.7d and 2.10 also show that our network preserves the timing of the target sounds and can silence out noise outside the target sound duration.

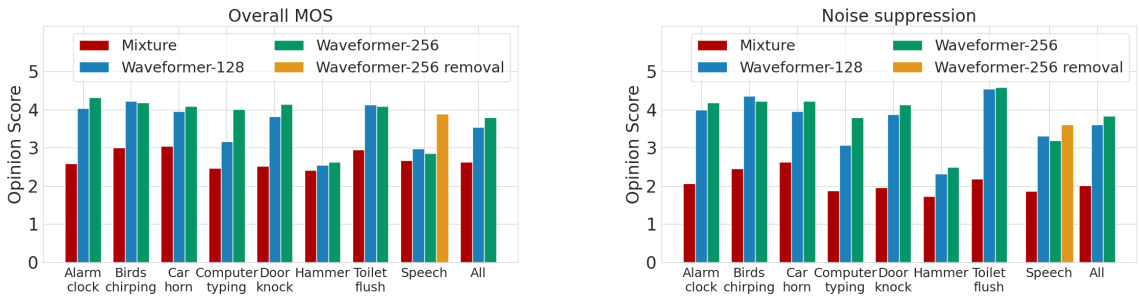
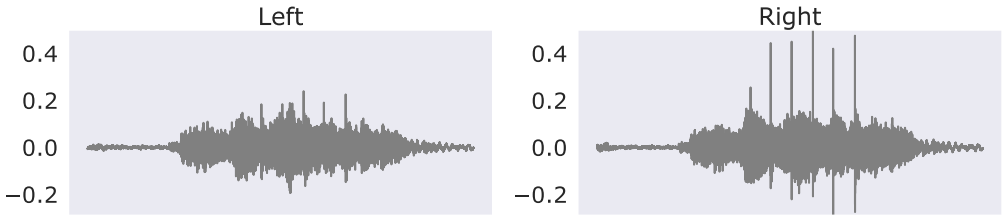
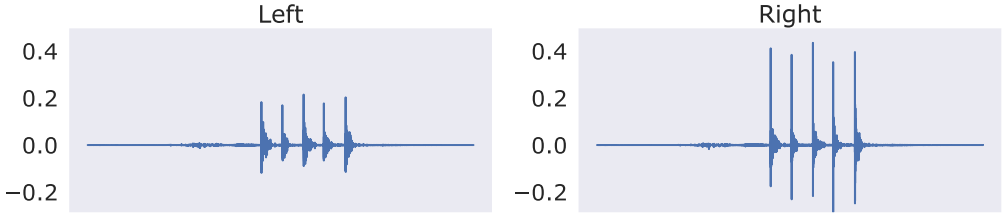


Figure 2.9: In-the-wild evaluation results for (a) mean opinion score (MOS) and (b) noise suppression across various classes that occurred in real-world data collection.



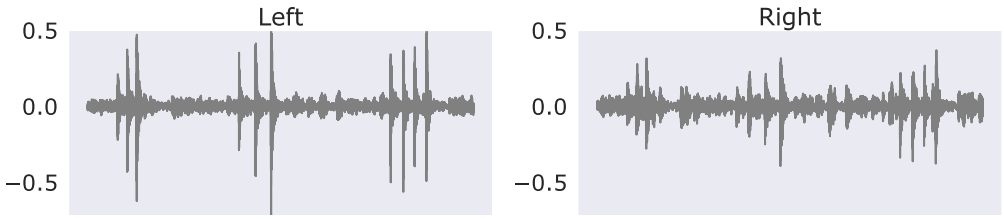
(a) Binaural input of door knock in the presence of toilet flush.



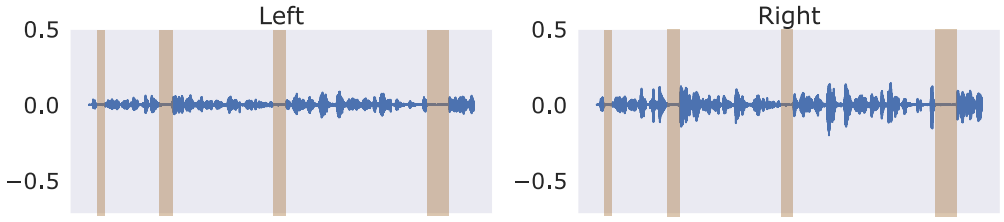
(b) Binaural output with door knock extracted.

Figure 2.10: Qualitative result with a real-world recording.

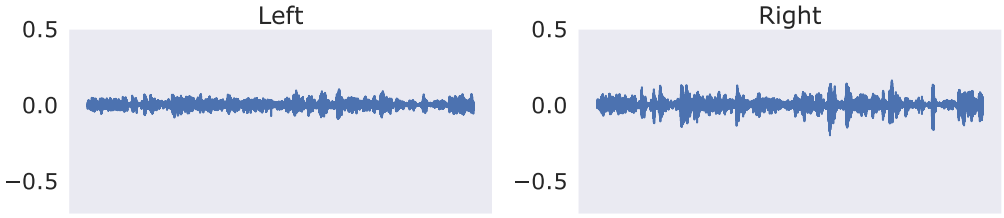
The results also offer interesting insights at a per-class level. In general, the 128-channel model performs only slightly worse than the 256-channel model for almost all classes, except for the "Computer typing" class, where the gap in the overall MOS between the two models is almost 0.84 MOS points. This is likely due to a particularly noisy recording taken near a running generator, where the 128-channel model created faint, unpleasant artifacts that were not observed with the 256-channel model. However, both models performed poorly in the "Hammer" class, where the target hammer sound was recorded in the presence of interfering music. Although the network correctly silenced the time segments that did not contain the hammer sounds, there was a noticeable residue from the music when there was a hammer sound, which the listeners found intrusive. Another important finding from the study is the significant improvement obtained by removing interfering signals from the input recording when the target is speech. By removing short-length sounds such as door knocks from the recorded signal instead of extracting the speech directly (see



(a) Binaural input of speech recorded with door knock



(b) Binaural output with speech extracted



(c) Binaural output with door knock removed

Figure 2.11: Extracting speech as a target here causes momentary periods of excessive signal attenuation (highlighted in b) as the network tries to remove door knocks and background sounds. However, if we extract and then subtract door knock sounds, the background noise is still faintly present, and the resulting signal sounds less harsh.

Fig. 2.11), we were able to increase the overall MOS by 0.91 points. Finally, it's worth noting that these in-the-wild results were obtained from the models trained solely on synthesized data, without any training on data collected from our hardware or for the participants.

2.4.3 Evaluating user-perceived spatial cues

We present experiments conducted in five ordinary, reverberant rooms to evaluate the ability of our design to preserve or recover user-perceived spatial cues. As with the in-the-wild evaluation, our training data had no samples either from our hardware or the tested real-world environments.

Data collection. We collected real-world audio recordings of our target sounds from known directions. To achieve this, five participants (3 male, 2 female) were fitted with binaural microphones and seated on a rotating chair positioned at the center of a large, printed semicircular protractor measuring 70×36 inches, as

shown in Fig. 2.12. The protractor was lined at regular 22.5° intervals (nine lines total) for precise rotational measurement. A Sony SRS-XB10 loudspeaker was placed on a fixed tripod at the 90° line of the protractor to emit different sound signals. To control the angle-of-arrival of the sound signal relative to the listener, the participants were asked to rotate the chair and align themselves with one of the protractor's lines.

The data was collected in 9 stages. In each stage, the user is rotated towards a different angle. The first stage starts with the participant facing the 180° line. After completion of each stage, the participant rotates 22.5° clockwise to the next marked angle. At each stage, the loudspeaker plays four 5-second audio samples: (1) white noise, (2-3) two test samples belonging to the target sound classes, and (4) a test sample belonging to the interfering other sound classes. Across all stages of data collection, the chosen audio samples comprise exactly 9 test samples from 9 distinct interfering other sound classes and 6 test samples from 6 distinct target sound classes. Notably, each test sample from the target classes is recorded for 3 different relative angles.

Evaluation procedure. Since our goal is to develop a system that accurately preserves the spatial cues perceived by human listeners, we design a user study to compute the perceived angle-of-arrival for the target binaural sounds output by our system. To this end, based on the collected audio recordings, we first create sound mixtures by sampling two audio clips from the target classes, and 1-2 clips from the interfering other classes. The mixtures are generated using Scaper. We choose the reference loudness of the background to be -50 LUFS, and we set the SNR of the target class sounds to 15-25 dB and that of the interfering other class sounds to 0-10 dB. We process each mixture by choosing a target class and running the mixture through our network.

We play the recordings of the individual clean target sounds with no interference, as well as the network output samples estimating these target sounds from the created mixtures, to the same set of participants via a pair of binaural earphones. Since the perceived spatial cues rely heavily on anthropometric features, all the sound signals played to a given participant originated from the binaural data obtained from that same participant in the data collection step. Prior to listening to each sample, participants are informed of the target class they should be localizing. After listening, they are asked to predict the direction of the sound source. To prevent the participants from associating each output sample with its corresponding individually-recorded target sound, the samples are played in a random order. To help the participants establish an

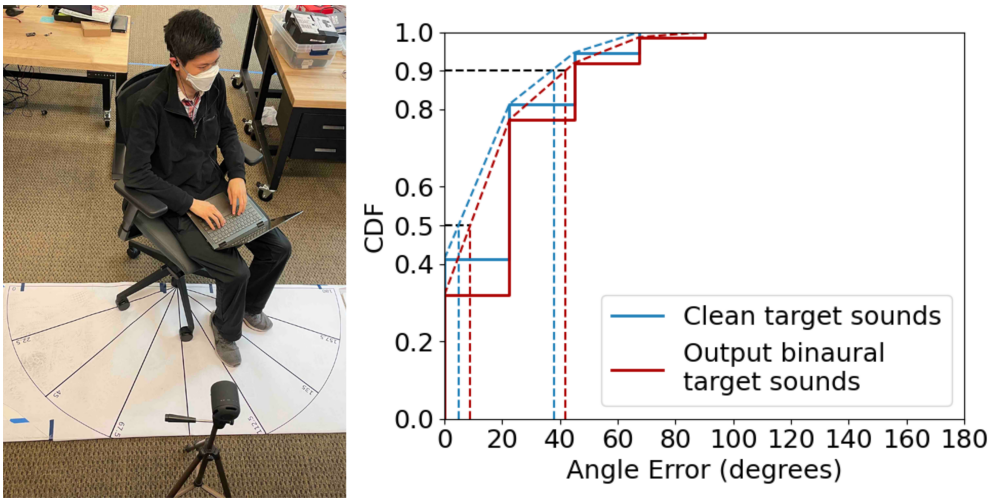
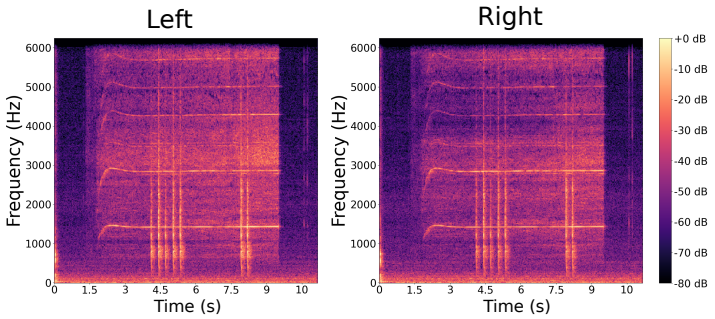


Figure 2.12: Spatial cue evaluation. (left) the evaluation setup, and (right) the CDF of the error between the ground truth source direction and the user-perceived source direction after listening to the isolated clean target sounds as well as network output binaural target sounds. The dashed lines are interpolated CDFs used to compute the interpolated median and 90th percentile error.

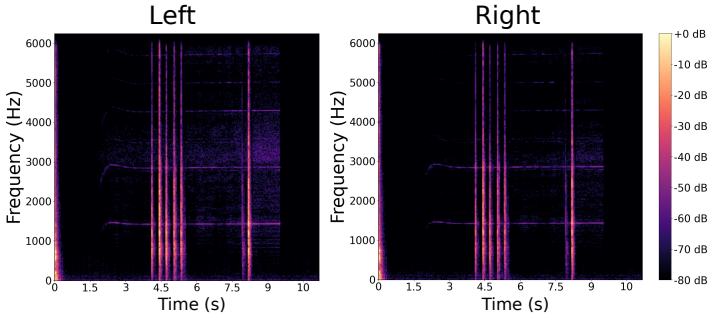
orientation reference, we play back the white noise samples for each angle in the increasing order at the start of the evaluation. Additionally, in cases of uncertainty between two specific source angles, the participants are allowed to re-listen to the white noise samples recorded for these angles. The study lasted around 20 minutes per user.

Results. We compare the errors between the ground truth source directions and the users' perceived arrival directions obtained for both the clean interference-free target sound recordings as well as the binaural target sound signals generated by our system for the mixture signal input. Our findings, as illustrated in Fig. 2.12, show that the mean angle error slightly increases from 18° to 23.25°. Additionally, we observe that the interpolated 50th and 90th percentile errors also increase marginally from 5° to 9° and from 38° to 42°, respectively. This demonstrates that our model preserves the spatial cues of the target sounds in its output and has a negligible impact on how users perceive the source directions.

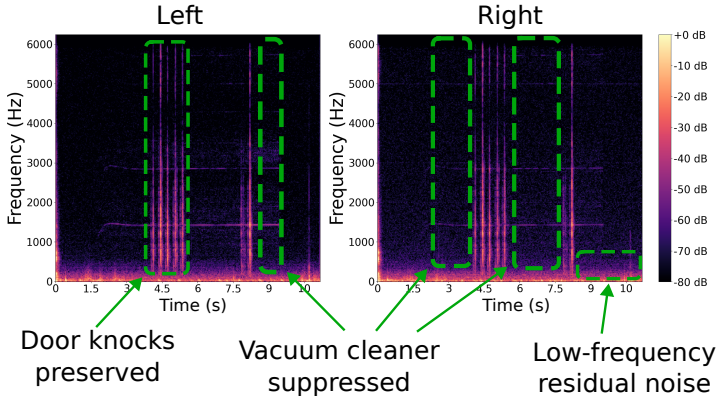
So far, we have treated semantic hearing and active noise cancellation as two separate systems that function independently. In practice, however, the end-to-end system requires a few additional considerations. Firstly, many active noise cancellation systems rely on a recorded signal inside the ear cup to adaptively silence the noise signals and achieve adaptive noise cancelation. Hence, the audio we play back to perform semantic hearing may influence the noise cancellation algorithm. Secondly, active noise cancellation systems are not perfect, and they may still let some sounds through. To address these concerns, we record data



(a) Sound recorded outside the headphone cups.



(b) Semantic hearing output played by headphones.



(c) Sound recorded inside the headphone cups.

Figure 2.13: Spectrograms of binaural recordings showing results from our end-to-end experiment with a wearable headset. Here, we extract door knock sounds in an environment with a nearby active vacuum cleaner.

while a user is utilizing our end-to-end system in real time. The user wears a pair of Sony WH-1000XM4 headphones with active noise cancellation enabled. In addition to the outer microphones used to capture external sounds to process, they also wear binaural microphones inside the earcups to record the sound produced by the active noise cancellation and semantic hearing systems together, i.e. as heard by the user. The user chooses to listen to the sound of door knocks as a vacuum cleaner is turned on nearby. For this

experiment alone, we run our semantic hearing algorithm on the audio recorded from the outer microphones on a laptop with an Intel Core i5 CPU. The processed audio is played back through the headphones.

Fig. 2.13(a)-(c) shows the spectrograms for three binaural signals: the signal recorded at the outer microphone, the signal played through the headphones, and the signal recorded inside the earcups. We demonstrate that while the recordings from the inside earcups are slightly noisier, we clearly see that the system can suppress the unwanted sounds (vacuum cleaner), while preserving the target sounds (door knocks). This demonstrates the feasibility that such a system can coexist with active noise cancellation systems. We note that to mitigate residual noises, the semantic hearing subsystem may have to integrate the residual audio from noise cancelling headphones to adapt the playback signal to the residual noise as well. However, this comes with stricter latency requirements and thus we leave it for future work.

2.4.4 Benchmarking the neural network

In-the-wild evaluation with human evaluators is closest to real-world use. It is however hard to objectively compare different models due to the lack of ground-truth signals, as well as due to the challenges in obtaining a large amount of test data necessary for the statistical significance of smaller performance gaps. To address these practical limitations, we also evaluate our model on an extensive reverberant binaural test-set comprising 10000 mixture and ground-truth pairs. We synthesized the benchmarking dataset to mimic real-world situations.

Table 2.3: Performance and efficiency comparison of different binaural target sound extraction frameworks and mask estimation architectures on a large test dataset across 20 target classes.

Binaural framework	Mask estimator	Params (M)	SI-SNRi (dB)	ΔITD (μs)	ΔILD (dB)	Runtime (ms)
Dual-ch	Ours ($D = 128$)	0.52	7.17	87.77	0.88	6.56
	Ours ($D = 256$)	1.74	7.41	85.16	0.87	12.54
Parallel	Ours ($D = 128$)	0.86	7.24	81.72	1.08	13.35
	Conv-TasNet	2.33	4.43	670.05	-	15.58
Single-ch	Ours ($D = 256$)	1.68	7.43	79.70	1.32	22.19
	Waveformer ($D = 256$)	1.69	7.37	85.33	1.27	25.85

To evaluate the performance of our binaural extraction model, as shown in Table 2.6, we compare the following three binaural target sound extraction frameworks.

- **Dual-ch.** This is the dual-channel architecture we proposed in §2.4 for efficient binaural target sound

extraction. In this framework, the binaural signal is converted into a combined latent space representation before the mask estimation. Since both left and right channels are combined into a common representation, a single instance of the mask estimation network is used for estimating the mask corresponding to the target sound. We consider our mask estimation architecture with both $D = 128$ and $D = 256$.

• **Parallel.** This is the binaural framework proposed in Han et al. [2020] that implements parallel processing of the left and right channels, along with some cross-communication between channels. The binaural framework in Han et al. [2020] is originally proposed for binaural speech separation. We implemented this framework for both our mask estimation network with $D = 128$ and Conv-TasNet Luo and Mesgarani [2019a]. We include Conv-TasNet as it is one of the most widely used signal enhancement model architectures. We choose a configuration of Conv-TasNet that resulted in similar runtime to that of our model and trained both models with our training dataset.

• **Single-ch.** In addition to the above two binaural extraction frameworks, we also evaluate and compare the performance with a single-channel extraction baseline. Since the target sound extraction models we consider are sample-aligned, models trained with monaural inputs and outputs can be independently applied to the left and right channels. Similar to the *Parallel* case, this also involves two instances of the mask estimation network. However, by contrast, the model parameters applied to the left and right channels are the same and there is no cross-communication between the channels. We implement the best configuration of our model ($D = 256$) so that this serves as a strong baseline.

For each model, we compare the performance in terms of the signal quality, the accuracy in spatial cues, and the on-device runtime requirement. We measure the signal quality using the scale-invariant signal-to-noise-ratio Roux et al. [2018] improvement (SI-SNRi) of the output compared to that of the mixture, computed with respect to the ground-truth. The SI-SNRi results are averaged over the entire testset, across the left and right channels. Following Han et al. [2020], the spatial cue accuracy is measured using the difference in the interaural time differences (ITDs) and interaural level differences (ILDs) between the output binaural signal and the ground-truth binaural signal, denoted as ΔITD and ΔILD. We compute ITD using cross-correlation, limiting them to ± 1 ms, as was done in May et al. [2011]. The model runtimes are measured on iPhone 11, by converting them to ONNX format Bai et al. [2019] and then executing them using ONNX Runtime for iOS. The runtimes are measured for computing a 10 ms output chunk averaged

over 100 runs. Therefore, the runtime must be less than 10 ms for deployment, which our dual-channel model with $D = 128$ meets.

In our experiments, we observed that the causal Conv-TasNet converges to the local minima of generating a constant zero signal when trained only with the SNR loss. This phenomenon is also observed in Veluri et al. [2023b], which suggested training Conv-TasNet with 90% SNR + 10% SI-SNR loss. The likely cause for this is, unlike the speech datasets that Conv-TasNet is originally designed for, sound datasets have a significant amount of silence, causing the Conv-TasNet optimization process to converge to generating a zero signal. On the other hand, using a loss of 90% SNR + 10% SI-SNR in the binaural case, caused one of the channels to output a very low-amplitude signal relative to the other channel as SI-SNR is insensitive to the signal gains. We confirmed that the signal is spectrally meaningful even though the magnitude is wrong. As a result, only SI-SNRi and ΔITD results are meaningful for the Conv-TasNet model. ΔILD computation resulted in infinity, so we omit it in our table.

In Table. 2.6, we observe that the dual-channel framework is competitive with the parallel and single-channel frameworks in terms of SI-SNRi, while outperforming in ΔILD. With regard to ΔITD, it resulted in a very marginal increase. These results intuitively make sense because the dual-channel framework has a sample-aligned common representation for both left and right channels. As a result, it can maintain the relative amplitudes of the left and right channels. On the other hand, the parallel and single-channel frameworks have separate branches that independently process different channels, facilitating maintaining the sample alignment with the respective channels. This phenomenon is more notable for the single-channel framework, where the SI-SNRi and ΔITD are promising but the ΔILD is poor, as there is no cross-communication between the left and right channel processings. We note that our dual-channel framework requires only a little more than 50% of the runtime required by their parallel or single-channel counterparts, making it a good practical choice for our semantic hearing system. Finally, we note that our dual-channel framework uses 240 MFLOPS, while vanilla Waveformer uses 357 MFLOPS across the two microphones.

For our causal model, the receptive field is exclusively the past audio. Hence, it has no effect on the algorithmic latency. The algorithmic latency of our model is the sum of chunk size, KL, and the lookahead of the input convolution, L, where L is the stride of the input convolution. Table 2.6 uses stride $L = 32$ samples and $K = 13$, resulting in a chunk size of $KL = 416$ samples and lookahead $L = 32$ samples. This

is equivalent to 9.4 ms and 0.7 ms, respectively. Table 2.4 shows the performance of our binaural model with various chunk sizes to understand the effect of algorithmic latency on performance. The results show that our model achieves reasonable performance with an algorithmic latency as low as 1.4 ms. Thus with ASIC implementations, such as those in hearing aids, we could envision ultra-low-latency semantic hearing systems.

During our in-the-wild evaluations, users freely moved their heads and encountered mobile sources (eg. sirens). The model also performed robustly without glitches during evaluations by human testers. The model adapted quickly to relative motion because it outputs small chunks (<10ms) while updating its internal state. The model can also utilize spatial positions in the trajectory that have better level differences between L and R channels. In addition to that qualitative evaluation, Table 2.5 provides a quantitative comparison of the performance for different amounts of relative angular motion between the listener and sound sources. For this comparison, we use the dual-ch model with $D = 256$ dimensions. We simulate motion using Steam Audio SDK SDK [2023], which simulates binaural motion given an HRTF file in the SimpleFreeFieldHRIR format sim [2023]. We performed controlled experiments with different angular velocities in both anechoic and reverberant environments, with sources moving from a random position on an arc with the given angular velocity. We used the CIPIC Algazi et al. [2001] HRTF dataset for anechoic simulations and RRBRIR IoSR-Surrey [2016] BRIR dataset for reverberant simulations as they provide impulse responses in the SimpleFreeFieldHRIR format. We synthesize the binaural audio in frames of 1024 samples by convolving with an interpolated impulse response using bilinear interpolation at every frame. Since the ILD and ITD are now time-varying, we compute the ΔILD and ΔITD in chunks of 250 ms, discarding any chunks where the clean signal is silent on both channels, and take the mean across the remaining chunks. We observed that in the presence of motion, SI-SNRi and ΔILD are marginally better as the model is able to better leverage the level differences between L and R at different relative angular positions while achieving lower ΔITD in the anechoic case and slightly higher ΔITD in reverberant scenarios.

2.5 Target speech hearing with noisy examples

In this section, we discuss how we can extend a framework similar to target sound extraction, to target speech hearing. The past decade has witnessed two key technological trends. First, there have been significant

Table 2.4: Effect of algorithmic latency on the performance. *Proposed system with end-to-end latency $\sim$20 ms.

Chunk size (samples)	Algorithmic latency (ms)	SI-SNRi (dB)
32	1.4	6.59
128	3.6	6.83
256	6.5	7.18
416*	10.1	7.42

Table 2.5: Comparison of performance in the presence of relative angular motion between listener and sound sources. Dual-ch model with $D = 256$ is used for this evaluation.

Angular velocity (°/s)	Reverb.	SI-SNRi (dB)	ΔITD (μs)	ΔILD (dB)
30	No	7.95	34.26	0.58
	Yes	7.88	103.45	0.43
60	No	7.91	49.49	0.57
	Yes	7.98	98.23	0.49
90	No	7.87	58.67	0.54
	Yes	8.00	99.83	0.43

advances in noise-canceling headsets and earbuds capable of better suppressing all environmental sounds air [2023]; Headphonesty [2022]; Review [2023]. Second, deep learning is enabling promising human-like intelligence across various domains Bubeck et al. [2023]; Jiang et al. [2023b]. These two trends present opportunities for creating the future of intelligent hearables, with real-world capabilities that so far have been in the realm of science fiction. In this paper, we explore a novel capability for hearables — *target speech hearing* — that allows users to choose to hear target speakers based on user-selected target speaker characteristics, such as speech traits.

Specifically, we explore the following question: can we look at a target speaker within a crowd just once, extract their unique speech traits, and subsequently employ these traits to exclusively listen to that speaker, while filtering out other voices and background noise? A positive answer could enable novel hearable applications that are currently not possible. For example, imagine a scenario in which a user seeks to hear only the tour guide's narration during a guided tour amidst the surrounding chatter and ambient noise while enjoying the tour sights. Alternatively, picture a leisurely stroll with a colleague along a cacophonous street, wanting to hear only their conversation and block out other sounds. Or think about being on a crowded bus, desiring to hear your friend talk while simultaneously gazing out of the window. While

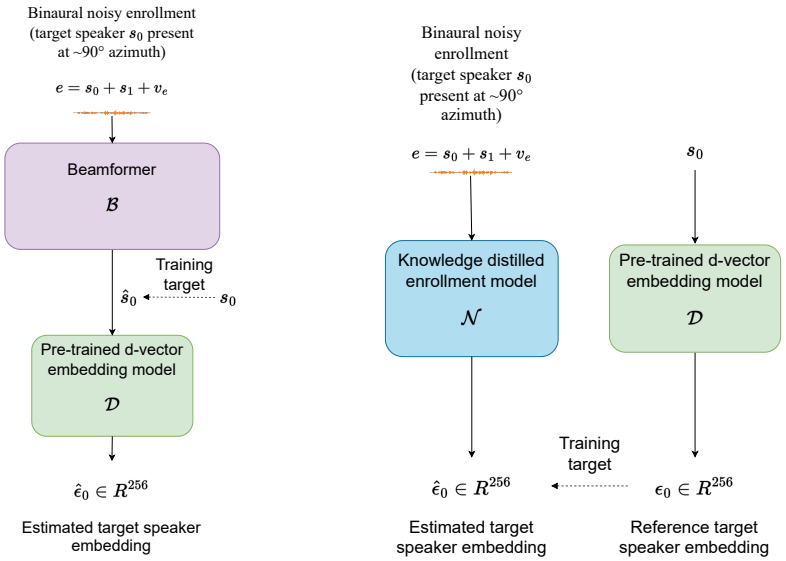


Figure 2.14: Target speech hearing with noisy enrollments. In (a) and (b), we propose two approaches for performing noisy enrollment, assuming the target speaker's azimuthal angle is approximately equal to 90°. (a) shows the beamformer-based approach, where a beamformer is trained to estimate the target speech signal from the noisy enrollment. The estimated target speech signal is then used to estimate the target speaker's embedding. (b) shows the knowledge-distillation approach, where an enrollment model is trained to estimate the reference d-vector embedding of the target speaker present at $\sim$ 90° azimuth. Once the speaker embedding estimated with one of the two approaches, we could perform target speech hearing in real-time, similar to the approach detailed in Fig. 2.6.

today's noise-canceling headphones have seen significant improvements in canceling out *all* sounds, they cannot selectively pick speakers based on their speech traits. These use cases, however, require not only using noise-canceling headsets to remove all sounds but also playing only the target speech back into the hearables.

The latter, which we call target speech hearing, is a new capability for general-purpose hearable devices. Existing deep learning approaches for the problem of target speech extraction require prior clean audio examples of the target speaker Zmolikova et al. [2023]. These clean examples are utilized by a neural network to learn the characteristics of the target speaker, which are subsequently employed to separate their speech from that of other concurrent speakers. The challenge lies in the fact that this problem formulation does not align well with our target hearable application domain. Specifically, in all the previously described use cases, obtaining a clean example signal of the target speaker (e.g., tour guide) is difficult since the target speaker may always be in a noisy environment, with interference from other speakers.

Providing clean target speaker examples for enrollment is essentially a user interface problem, and hence requires the design of an intelligent hearable system that takes into account the constraints of a user-friendly interface. In this paper, we introduce the concept of target speech hearing on hearable devices with noisy examples. To achieve this, rather than expecting users to collect input examples of the target speaker in a noise-free environment in the absence of any other speakers, we show for the first time how one can enable target speech extraction using noisy binaural enrollments in the presence of other concurrent interfering speakers.

The wearer looks at the target speaker for a few seconds and captures binaural audio, using two microphones, one at each ear. Since during this short enrollment phase, the wearer is looking in the direction of the target, the signal corresponding to the target speaker is aligned across the two binaural microphones, while the other interfering speakers are likely to be in a different direction and are therefore not aligned. We employ a neural network to learn the characteristics of the target speaker using this sample-aligned binaural signal and separate it from the interfering speaker using direction information. Once we have learnt the characteristics of the target speaker (i.e., target speaker embedding vector) using these noisy binaural enrollments, we subsequently input the embedding vector into a different neural network to extract the target speech from a cacophony of speakers. The advantage of our approach is that the wearer only needs to look at the target speaker for a few seconds during which we enroll the target speaker. Subsequently, the wearer can look in any direction, move their head, or walk around while still hearing the target speaker.[2]

To make this idea practical, we make multiple contributions:

• **Enrollment networks with noisy examples.** We design and compare two different enrollment networks — a beamformer network and a knowledge distillation network (see §2.5.2) — to effectively generate a speaker embedding vector that captures the traits of the target speaker using the short binaural noisy example.

• **Real-time embedded target speech hearing network.** We use the generated embedding to subsequently extract the target's speech using an optimized network that runs in real-time on an embedded IoT CPU. To do this, we start with the state-of-the-art speech separation network, TFGridNet Wang et al. [2023d], which cannot run in real-time on our embedded device. We then introduce various model and system-level

[2]In contrast, directional hearing Wang et al. [2022a] focuses on speech from a specific direction. However, this approach is not well-suited to our application scenarios, as users do not continuously look at the target speaker, the target speaker may have long pauses in their speech making continuous direction tracking challenging, and the direction can change as they or the user move their head to look elsewhere (e.g., tour sights).

optimizations to achieve a light-weight target speech hearing network that runs in real-time on embedded CPUs.

• **Generalization to real-world multipath, HRTF and mobility.** We present a training methodology that uses only synthetic data and yet allows our system to generalize to real-world unseen target and interfering speakers and their head-related transfer functions (HRTFs). Further, we explicitly train with multipath to generalize to both indoor and outdoor environments. We also introduce a fine-tuning mechanism that addresses moving sources and sudden changes in the listener's head orientation (upto 90°/s angular velocity). This also allows the system to handle up to 18° error in the listener's head orientation during enrollment (see §2.5.3).

We build an end-to-end hardware system that integrates a noise-canceling headset (Sony WH-1000XM4), a pair of binaural microphones (Sonic Presence SP15C) with our real-time target speech hearing network running on an embedded IoT CPU (Orange Pi 5B). The embedded device reads audio chunks from the microphones, which we process on-device and play back to the headset. Our average model inference time was 6.2 ms to process 8 ms audio chunks, making it a real-time system with a total end-to-end latency of 18.24 ms. Our results are as follows.

• Compared to clean example enrollments, the beamformer network for noisy example enrollments resulted in 2.9 dB performance drop. In contrast, the knowledge distillation network resulted in only a 0.4 dB drop in performance compared to clean examples (see §2.4.4), while using only 1-4 second noisy enrollments.

• Our system generalized to 9 real-world settings that span different motion scenarios, indoor and outdoor environments as well as different wearer postures with 8 participants using our hardware. Our design does not require any training data collection with our hearable hardware.

• In a user study with 21 participants who spent over 420 minutes rating the target-speaker output by our hardware system from real-world indoor and outdoor environments, our system achieved a higher mean opinion score and interference removal for the target speaker than the raw unprocessed input.

• Across nine participants who compared three interfaces for noisy enrollments — push button on headphone, touchpad on headphone, and virtual button on a smartphone — participants expressed preference for the push button because of its good haptic feedback.

Imbued with embedded intelligence, our work envisions hearables that allow wearers to manipulate their acoustic surroundings in real-time to customize their auditory experience based on user-defined characteristics like speech traits. By open sourcing the code and datasets, our work may help further future research among HCI and machine learning researchers on designing algorithms and systems around target speech hearing.

2.5.1 Look Once to Hear

Our key observation is that for hearable applications of deep learning-based target speech extraction Žmolíková et al. [2019], it is often impractical to obtain a clean speech sample of the target speaker. In this work, we propose a *target speech hearing (TSH)* system suitable for binaural hearables applications that provides an interface for noisy in-the-wild speech samples, which we refer to as *noisy enrollments*. A noisy enrollment of a speaker of interest would contain two kinds of noise: uncorrelated background noise, and interfering speech. While the background noise can be suppressed with existing methods Hu et al. [2020], it is challenging to disambiguate and suppress interfering speech without suppressing the target speech itself, especially when the number of speakers in the scene could be arbitrary. More fundamentally, in a mixture of multiple speakers, it is challenging to know which of them is the intended target speaker.

Our system achieves this disambiguation by leveraging the beamforming capability of binaural hearables. Assuming that the listener would be looking at the target speaker at least for a few seconds, we propose that the listener could use this phase to *enroll* the speaker they want to focus on by letting the hearable know through on-device haptic control or a button click on the phone application. During this phase, since the direct path of the target speaker is equidistant from both ears of the binaural hearable, the application could disambiguate between target and inferring speakers to obtain a representation of the target speaker.

Let $e(t') \in R^2$ be the input binaural signal received by the binaural hearable during the enrollment phase, and $x(t) \in R^2$ be the input binaural signal received during TSH phase, where t' and t corresponds to the time during the enrollment phase and TSH phase, respectively. Then these signals could be decomposed into their component signals:

$$e(t') = s_0(t') + \Sigma_{k=1}^{m} s_{ek}(t') + v_e(t') \tag{2.7}$$

$$x(t) = s_0(t) + \Sigma_{k=1}^{n} s_k(t) + v(t) \tag{2.8}$$

Here, $s_0 \in R^2$ corresponds to the target speaker, $s_{e1}, ..., s_{em} \in R^2$ correspond to interfering speakers during the enrollment phase, and $s_1, ..., s_n$ correspond to interfering speakers during the TSH phase. Note that the interfering speakers can be the same or different during the two phases. $v_e(t')$ and $v(t)$ represent background noises in the respective phases. Additionally, let θ_0 represent the azimuthal angle of the target speaker, relative to the listener. During the enrollment phase, to achieve disambiguation of the target speaker in the noisy enrollment signal, since the user looks in the direction of the target speaker, we can assume that: $\theta_0(t') \sim \frac{\pi}{2}$, where the x-axis is assumed to pass from the listener's left to right ear with the midpoint as the origin. We then formulate the TSH problem as a two-step process:

$$\hat{\epsilon}_0 = \mathcal{N}(e(t')|\theta_0(t') \sim \frac{\pi}{2}) \tag{2.9}$$

$$\hat{s}_0(t) = \mathcal{T}(x(t), \hat{\epsilon}_0) \tag{2.10}$$

Here, $\hat{\epsilon}_0$ corresponds to the target speaker representation computed from the noisy enrollment signal $e(t')$, $\mathcal{N}$ is the neural network estimating the target speaker's representation and $\mathcal{T}$ is the real-time causal target speech hearing network that can run on an embedded device. $\mathcal{T}$ is the combination of mask estimator, convolution and transposed convolution modules (Fig. 2.6a), with the exception that we provide the speaker embedding directly, instead of the one-hot vector q. In the following subsections, we explain in detail, different architectures we explored for both the enrollment phase and TSH phase.

2.5.2 Enrollment interface network

The quality of the target speech extracted by the target speech hearing network, $\mathcal{T}$, has a critical dependence on the discriminative quality of the speaker representation, ϵ_0, provided to it. In order to robustly handle various speech characteristics, we leverage the speaker representations computed by large-scale pre-trained models such as Wan et al. [2020]; Koluguri et al. [2021]. In this work, we use the open-source implementation of Wan et al. [2020] in the Resemblyzer project Resemble-Ai [2019]. Given a clean speech utterance of a speaker $s_i(t')$, Resemble-Ai [2019] uses a long short-term memory (LSTM) network, $\mathcal{D}$, to map the utterance to a unit length 256-dimensional vector $\mathcal{D}(s_i(t')) = \epsilon_i$, where $\epsilon_i \in R^{256}$ and $||\epsilon_i||_2 = 1$, referred

to as a *d-vector embedding*. During the training phase, the LSTM model computes d-vectors optimized such that embedding corresponding to an utterance of a speaker is closest to the centroid of embeddings of all other utterances of the same speaker. This is done while simultaneously maximizing the distance from the centroids of all other speakers in the large-scale speech database used as the training set. In this work, we use d-vector embeddings as reference speaker representations that the noisy enrollment network $\mathcal{N}$ should predict using two approaches.

Noisy enrollment with beamforming. We note that the d-vector embedding of the target speaker can be obtained with its clean speech example as $\epsilon_0 = \mathcal{D}(s_0(t'))$. If we could estimate the clean speech of the target speaker, provided that the target speaker is present at the azimuthal angle $\theta_0 \sim \frac{\pi}{2}$, we could estimate the d-vector embedding corresponding to the target speaker. Essentially, this is equivalent to beamforming with direction input steered towards the azimuthal angle equal to $\frac{\pi}{2}$. In this work, we follow the *delay and process* approach proposed in several beamforming works Jenrungrot et al. [2020]; Chatterjee et al. [2022]; Wang et al. [2022a], where given a target direction and a reference microphone, inputs from other microphones are delayed according to the time it takes for the direct path from the given direction to reach them relative to the reference microphone. In this case, since we assume the direct path is equidistant from both left and right microphones, processing the raw inputs is sufficient to obtain the target speaker. Assuming the beamforming network is represented as $\mathcal{B}$, the process of noisy enrollment with beamforming could be written as:

$$\hat{s}_0(t') = \mathcal{B}(e(t')|\theta_0 \sim \frac{\pi}{2})$$

$$\hat{\epsilon}_0 = \mathcal{D}(\hat{s}_0(t'))$$

In this work we use the state-of-the-art speech separation architecture TFGridNet Wang et al. [2023d] as our beamforming architecture $\mathcal{B}$. Since enrollment is a one-time operation that does not need to be performed on-device, we could use the original non-causal implementation of the TFGridNet Wang et al. [2023d] available in the ESPNetLu et al. [2022] framework. Following the notation in Wang et al. [2023d], we used the configuration: $D = 64$, $B = 3$, $H = 64$, $I = 4$, $J = 1$, $L = 4$ and $E = 8$ with short-time fourier transform (STFT) window size set to 128 and hop size set to 64.

Noisy enrollment with knowledge distillation. Conversely, we could consider this problem as the

noisy enrollment network, $\mathcal{N}$, directly computing the estimated d-vector embedding of the target speaker, $\hat{\epsilon}_0$, given the noisy speech. This would however require us to use a resource-intensive training process like the one proposed in Wan et al. [2020]. To do this efficiently, we train the enrollment network $\mathcal{N}$, using knowledge distillation Hinton et al. [2015]; Asami et al. [2017], where the original d-vector model, $\mathcal{D}$, provides d-vector embeddings computed on clean target speech as ground-truth references. We note that during the training phase, we have access to clean target enrollment speech $s_0(t')$, but we do not assume this during inference. Here, we train the noisy enrollment network $\mathcal{N}$ to minimize the loss function $\mathcal{L}(\hat{\epsilon}_0, \epsilon_0)$:

$$\hat{\epsilon}_0 = \mathcal{N}(e(t')|\theta_0 \sim \frac{\pi}{2})$$

$$\epsilon_0 = \mathcal{D}(s_0(t'))$$

$$\mathcal{L}(\hat{\epsilon}_0, \epsilon_0) = cos(\angle(\hat{\epsilon}_0, \epsilon_0)) = \hat{\epsilon}_0 \cdot \epsilon_0$$

To make both our noisy enrollment approaches comparable, we use TFGridNet Wang et al. [2023d] with the same configuration as above, as the noisy enrollment network $\mathcal{N}$, in this approach as well. We modify the architecture to output 256-dimensional embedding instead of an audio waveform, as shown in Fig. 2.14b. The bulk of the TFGridNet architecture computes a 64x65-dimensional representation for each audio chunk, which is then processed by a final convolutional layer followed by inverse-STFT (ISTFT) to compute the output waveform. For the purpose of noisy enrollment, we directly use 64x65-dimensional representation and reduce it using a linear layer to output the 256-dimensional representation for each enrollment audio chunk. We then average the 256-dimensional representations over all enrollment audio chunks to obtain the final 256-dimensional target speaker embedding.

2.5.3 Training for real-world generalization

This section describes the training approach for the TSH system, which is a generalization of the training approach for target sound extraction. In TSH system, we train both the enrollment network and target speech extraction network conditioned on speaker embeddings. But in the case of target sound extraction, we only need to train the target sound extraction network conditioned on one-hot labels.

We first train the enrollment networks to estimate d-vector embeddings. We then separately train the tar-

get speech hearing model while conditioning it with reference d-vector vector embeddings. This approach allows us to use the same target speech hearing model with any enrollment model that can estimate d-vector embeddings. We train these models with a training dataset that considers an accurate representation of real-world use cases of a target speech hearing system. Specifically, we consider variations in speech characteristics, acoustic transformations caused by physical multipath environments, acoustic transformations caused by the human head related transfer function (HRTF) and diverse background noise. We also consider the effects caused by motion of the speaker and noise with respect to the listener as an additional finetuning step. Below we explain the dataset details followed by the training process of the enrollment and target speech hearing networks.

Synthetic dataset. Each training sample in our dataset corresponds to an acoustic scene comprised of 2-3 speech samples and background noise. To create an acoustic scene, we first sample a 5 second background noise sample and then overlay the target speech and inferring speech at random start positions. For obtaining target and interfering speech, we randomly select 2-3 speakers from the LibriSpeech dataset Panayotov et al. [2015], and select a speech sample of length 2-5s for each speaker. The enrollment signals are generated using the same approach as well. We used the `train-clean-360` component of LibriSpeech dataset that comprises 360 hours of clean speech with 439 and 482 different female and male speakers, respectively. We further select random noise samples from WHAMR! dataset Wichern et al. [2019] comprising a database of audio samples of real-world noisy environments. These audio samples, however, do not contain the effects of real-world indoor environments and human heads, which we found is important for extracting natural-sounding audio.

Accounting for multipath and HRTF. To account for these effects, we convolve each of the speech samples and background noise with a binaural room-impulse-response (BRIR) that captures the acoustic transformations caused by a room as well as a user's head and torso. Let $h_{r,\theta,\phi}$ be a BRIR corresponding to the room and subject combination r, at azimuthal angle θ and polar angle ϕ with respect to the subject's head. Let $S_0(t) \in R$ and $S_1(t) \in R$ be two mono clean speech mixtures sampled from the LibriSpeech dataset, and $V(t)$ be noise sampled from the WHAMR! dataset. Then the binaural acoustic scene $x(t) \in R^2$ for this source mixture could be computed as:

$$x(t) = S_0(t) * h_{r,\theta_0,\phi_0} + S_1(t) * h_{r,\theta_1,\phi_1} + V(t) * h_{r,\theta_v,\phi_v} \tag{2.11}$$

It is to be noted that in each acoustic scene, room and subject configuration r, remains the same for all sources, but the angles with respect to the listener are arbitrary. To improve robustness to variations in rooms and subject, we aggregate BRIRs from 4 different datasets: CIPIC Algazi et al. [2001], RRBRIR IoSR-Surrey [2016], ASH-Listening-Set ShanonPearce [2022] and CATTRIR IoSR-Surrey [2023]. Of these, CIPIC dataset only comprises of impulse responses measured in an anechoic chamber and as a result, is devoid of any room characteristics. Combined, these datasets provided us with a total of 92 different room and subject configurations.

Training. To train the enrollment networks, we first generate the component speech utterances, as described above, with the constraint that target speaker's azimuthal angle $\theta_0 \sim \frac{\pi}{2}$. We train the beamformer-based enrollment network to predict target speech (Fig. 2.14a) with SNR loss. We train knowledge-distillation-based enrollment network to predict the d-vector embedding of the target speech (Fig. 2.14b) with cosine-similarity loss.

To train the target speech hearing (TSH) network, we also sample a random speech corresponding to the target speaker and convolve it with a BRIR corresponding to the same room and subject configuration. We input the TSH model with the acoustic scene and the d-vector embedding computing on the sample speech. We then optimize the TSH network, $\mathcal{T}$, to minimize the signal-to-noise ratio Roux et al. [2018] (SNR) loss between the estimated target speech and the ground-truth: -SNR($\hat{s}_0(t), s_0(t)$).

Finetuning for motion, error in the enrollment angle and real-world noise characteristics. In the dataset setup described above, we assumed a constant azimuthal angle for each source as time progressed. This means that sources are stationary with respect to the listener's orientation, and the enrollment angle is close to $\frac{\pi}{2}$ and does not change with time. These assumptions, however, are not true in the real world as sources could be moving, or there might be a rotation in the listener's head resulting in significant relative angular velocities. We handle relative motion and time-varying error in the enrollment angle with an additional finetuning step. During finetuning, we make the azimuthal and polar angle time-varying in 2.11.

2.5.4 In-the-wild Evaluation

We evaluate our system in previously unseen indoor and outdoor environments, with participants who are not in the training data. We recruited 8 individuals (5 male, 3 female) to collect data in different in-the-wild

scenarios using our hardware. We ask 3 participants at a time to collect noisy enrollment signals, as well as noisy real-world mixture audio, in different acoustic environments while they read random text. Among the three participants, one of them is designated as the wearer, while the other two are the speakers. To collect a noisy enrollment for a given target, the wearer looks at the target speaker as they read a text. This mimics the "Look Once" phase in the real-world use, where the listener would look at the target speaker. As the target speaker reads the text, background sounds and, in all but one case which had significant noise, speech from the other speaker make this enrollment signal noisy.

We record multiple noisy binaural audio clips while the target speaker reads a different text in the presence of other environmental sounds and speakers. Unlike the noisy enrollment signal, there is little control over these recordings, as the wearer and target speaker are free to move around and/or rotate their head. These recordings were also collected in different acoustic environments, including living spaces, busy streets, and in nature. They also contained settings where the listeners were in different postures, such as standing, sitting and laying down.

Evaluation procedure. Since the target speaker is speaking in the presence of interfering speakers and unknown noise, it is difficult to obtain the ground truth audio signal for our target speakers in the real world. So, we cannot rely on objective metrics to evaluate the system performance. Instead, we design a listening survey to allow human participants to rate the performance of our two enrollment methods on 15 different target speaker scenarios from the in-the-wild dataset we collected. To do this, we recruited 21 participants (13 male and 8 female with an average age of 30.4 years) to take our survey and give their opinion on our target speaker hearing system to obtain a mean opinion score (MOS). To do this, for each scenario, we first ask the users to try listening to a 5-second clean signal of the target speaker reading text collected in a quiet room. We then ask the participants to listen to the target speaker in 3 distinct audio clips: 1) the original, noisy recording of the target speech with interferers, 2) the output of our target speaker hearing network using the noisy knowledge distillation embeddings, and 3) the output of our target speaker hearing network using the noisy beamformer embeddings. The 3 clips are presented in random order.

After listening to the mixture and model output clips in a random order, we ask the participants to rate the target speech extraction quality by asking them the following questions:

- **Noise suppression**: *How INTRUSIVE/NOTICEABLE were the INTERFERING SPEAKERS and BACK-*

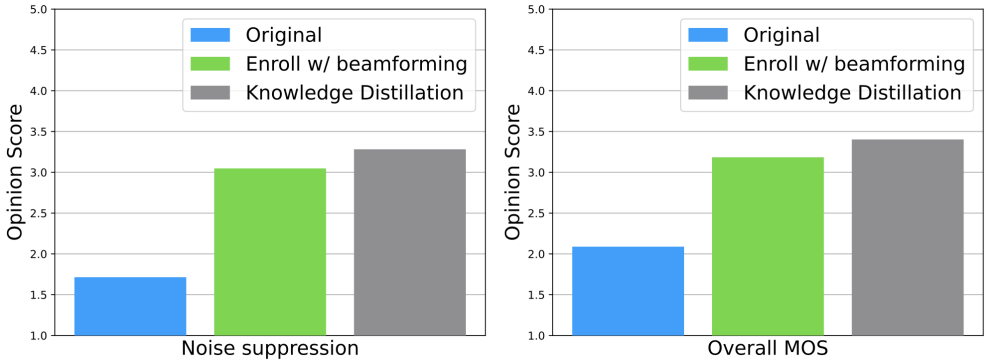


Figure 2.15: Subjective in-the-wild evaluations. (a) Mean opinion score for the noise suppression quality reported for the raw audio signal and the output using our two enrollment networks, and (b) overall reported mean opinion score. Paired t-tests between knowledge distillation and beamforming approaches resulted in p-values < 0.001.

GROUND NOISES? 1 - Very intrusive, 2 - Somewhat intrusive, 3 - Noticeable, but not intrusive, 4 - Slightly noticeable, 5 - Not noticeable

- **Overall MOS**: *If the goal is to focus on this target speaker, how was your OVERALL experience? 1 - Bad, 2 - Poor, 3 - Fair, 4 - Good, 5 - Excellent*

Results. Fig. 2.15 shows that our system can greatly suppress the background sounds and interfering speakers, as evidenced by the fact that that our beamforming and knowledge distillation enrollment networks increased the mean opinion score for the noise suppression task from 1.71 to 3.05 for the beamformer enrollment method and 3.28 for the knowledge distillation method (Fig. 2.15(a)). Our target speaker hearing framework was also able to improve the overall mean opinion score from 2.09 to 3.18 and 3.4, respectively (Fig. 2.15(b)).

2.5.5 Enrollment interface user study

We investigate two main questions: 1) What interface should the user interact with when they want to enroll the target speaker?, and 2) what enrollment duration do users find acceptable for such a system? As shown in Fig. 2.16, we integrate into our prototype with three different interfaces through which users can communicate their intention of enrolling a target speaker: 1) a virtual button on a smartphone application, 2) a push button on the headphone, and 3) a touch pad on the headphone. We evaluate four different possible enrollment durations: 2.5 s, 5 s, 7.5 s and 10 s.

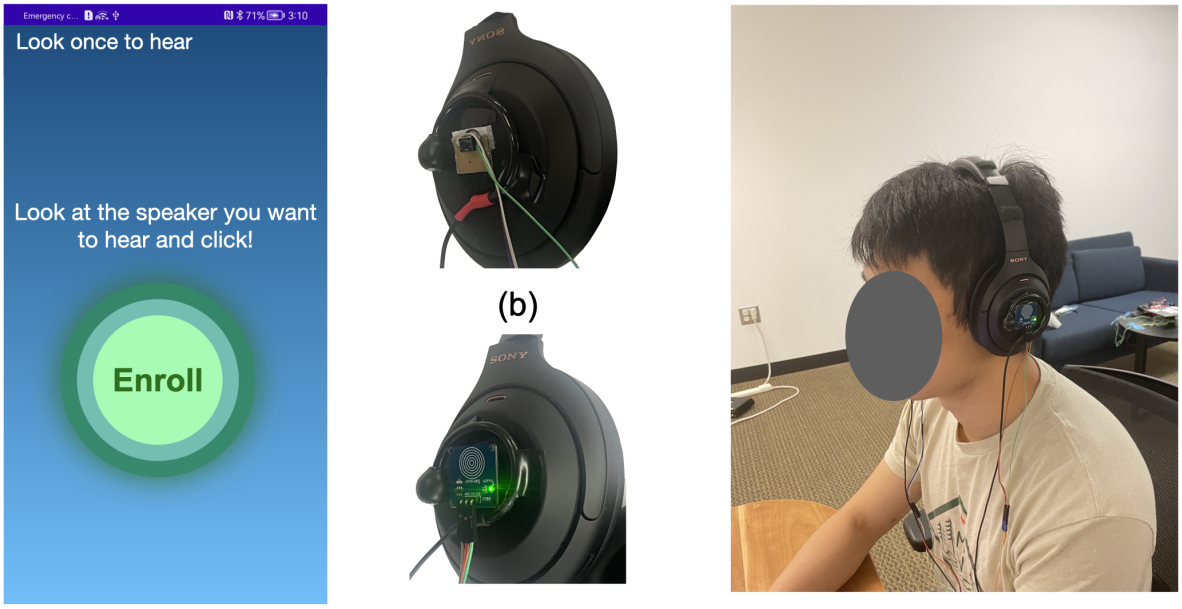


Figure 2.16: Proposed interfaces. (a) A smartphone app, (b) a push button, and (c) a touch pad. In (d), we see a participant wearing our prototype while conducting the user study.

Comparing user interfaces. We conducted a user study with 9 participants, where each participant first wore our device and sat on a chair as shown in Fig. 2.16(d). We placed a loudspeaker to the side of the wearer, which played a mixture of human speech from the LibriSpeech dataset and generic vacuum noise. A person sitting in front of the participant would then read a random text. We then asked the participants to use each of the three interfaces to signal to the system their intention to enroll the target speaker in front of them, while suppressing the interfering sounds emitted by the loudspeaker. When the users correctly interacted with the device to start enrollment, the headset would play a voice saying "Enrollment start". While enrolling the target speaker, the participants are asked to keep their head facing the target speaker, until the enrollment duration has passed and the enrollment is completely recorded, at which point a voice is played over the headphones, saying "Finished". The enrollment duration for all three interfaces was set to 5 seconds. After interacting with the three interfaces, we asked each participant to rate the three interfaces from 1-5 based on how likely they are to use that interface for this interaction. The results are shown in Fig. 2.17(a). Most participants showed a strong preference for the push button because of its good haptic feedback. All the participants showed the least preference for the smartphone since it required the extra

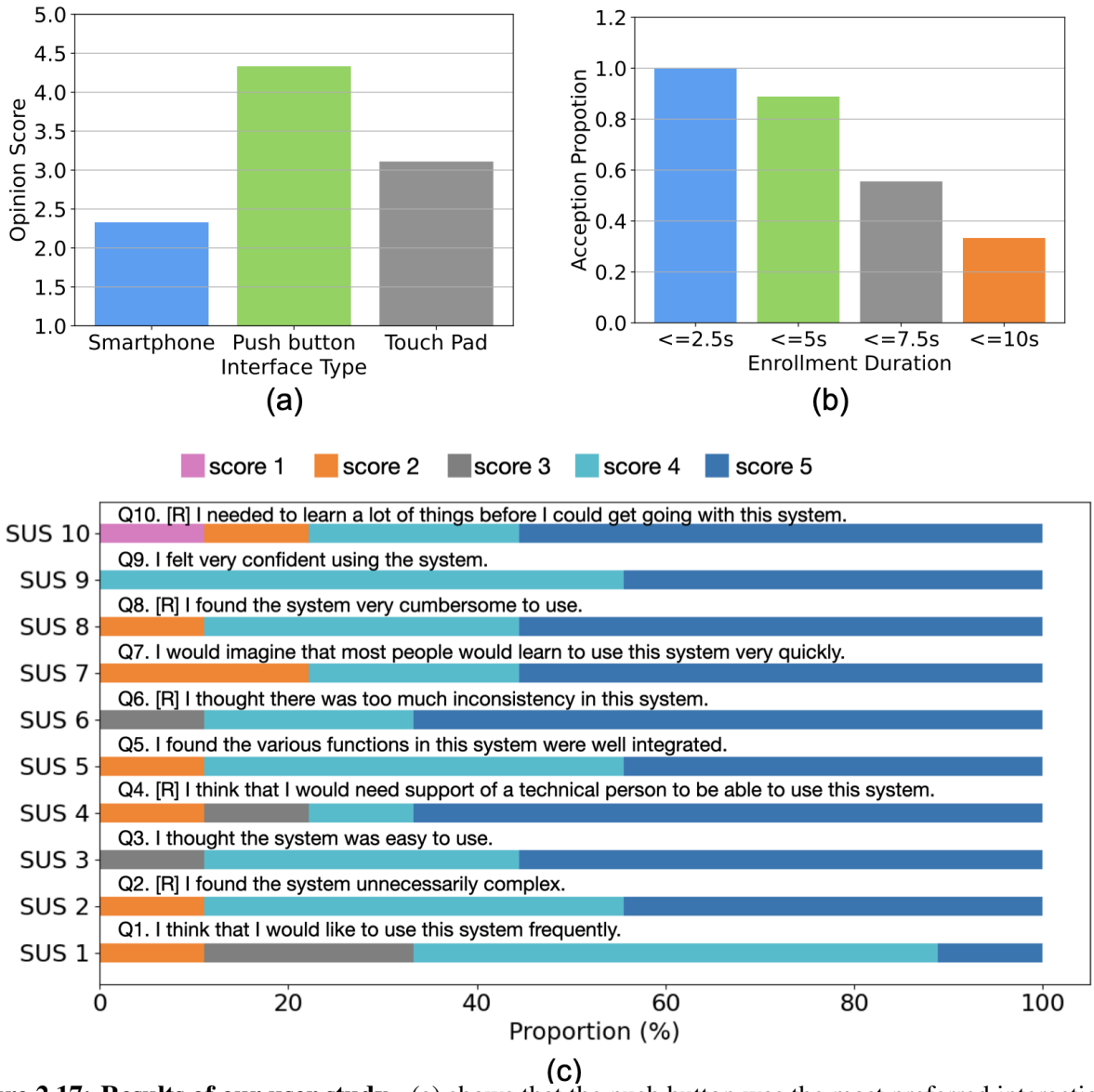


Figure 2.17: Results of our user study. (a) shows that the push button was the most preferred interaction method, while the smartphone app was the least preferred option. (b) shows the participant preferences with different enrollment duration. (c) shows the results of the SUS questionnaire, where we reverse the scale of negatively worded statements (Q2, Q4, Q6, Q8 and Q10) for easier visualization.

complexity of looking at the smartphone screen while simultaneously trying to face the target speaker.

Evaluating the enrollment duration. We then asked the participants to use their favorite interface from the previous study to explore their perspectives on a reasonable enrollment duration. Specifically, each participant performed enrollments with 4 different durations: 2.5s, 5s, 7.5s and 10s. The results in Fig. 2.17(b) show that 89% of the participants thought that 5 seconds was an acceptable duration for the

Table 2.6: Benchmarking results on the generated test set. Proposed noisy enrollment methods are evaluated with 3 different audio/speech processing architectures. Performance with clean enrollments is also provided for reference.

Enrollment network	d-vector similarity	Real-time TSH backbone	SI-SNRi (dB)	Params (M)	MACs (GMAC)
Clean	1.0	Streaming TFGridNet	7.40	2.04	4.63
		Waveformer	4.94	1.6	2.43
		DCCRN	6.71	5.54	6.6
Beamformer	0.74	Streaming TFGridNet	4.53		
		Waveformer	2.34	"	"
		DCCRN	4.34		
Knowledge distillation	0.85	Streaming TFGridNet	7.01		
		Waveformer	4.63	"	"
		DCCRN	6.16		

enrollment period.

Qualitative Results. Next, we asked each of the participants to fill out a System Usability Scale (SUS) questionnaire, which was developed in Brooke [1995]. The overall score was 80.8 ± 16.7, which suggests generally positive feedback on usability. The SUS results for each question are shown in Fig. 2.17(c), which correlates with being highly usable and acceptable by users, according to Bangor's Bangor et al. [2008] empirical evaluation. Finally, in addition to the previous studies, we also ask the subjective question: "Where do you see yourself using such a system?". Five of the participants mentioned using them in crowded scenarios and expressed similar applications to the response of one of the participants: "I'd like to use it in large social gatherings like conferences and lectures. I want to just talk with a specific people without being distracted by others or loud background noise". Five of the participants also mentioned that they were willing to use it in common public locations such as cafes, restaurants, on the street, karaoke and in large parties. Furthermore, one participant proposed that this technology might be useful for hearing aids. While all participants gave positive feedback and proposed useful potential applications, two of them also raised some limitations. One participant said, "In the real-world, I would also want to focus on a group of people instead of only one person". Another participant said "I think the headphone form factor is a bit obtrusive. A wireless earbud form-factor would be more socially acceptable."

2.6 Conclusion

The above works take an important first step towards realizing real-time programming of acoustic scenes on binaural hearable devices using the semantic description of sounds. In our work related to binaural target sound extraction, at its core are two key technical contributions: 1) the first binaural target sound extraction neural network. Our network can run in real-time, using 10 ms or less of audio blocks, while preserving the spatial information, and 2) a training methodology that allows our system to generalize to unseen real-world environments. In-the-wild experiments with participants show that our proof-of-concept hardware-software system can preserve the directions of the target sounds and separate these sounds in real-time from both the background noise and other sounds in the environment.

In the later work, we introduce the concept of target speech hearing using noisy examples on hearables that allows a user to focus on a specific speaker, given their speech characteristics, while reducing interference from other speakers and noise. We make three key technical contributions to achieve this new capability for hearables: 1) an enrollment interface that uses a noisy, binaural recording of the target speaker to generate a speaker embedding that captures that traits of the target speaker, 2) a real-time neural network that runs on embedded IoT CPU to extract the target speaker given the speaker embedding, and 3) a training methodology that uses synthetic data and yet allows our system to generalize to real-world unseen speakers, indoor and outdoor environments as well as support mobility. Our in-the-wild evaluations show generalization to real-world unseen indoor and outdoor environments.

Chapter 3

Spoken Language Modeling

Consider an interpreter in the United Nations, a mental health counselor, a TA in a course office hour or something much better than a karaoke machine, a personal band. A common task all of them do is to understand human voice, including all the emotion associated with it, and generate a continuous vocal or acoustic response. While this might be an oversimplification of the diverse roles humans are playing in different capacities, we hypothesize that any model that is intelligent enough to generate arbitrary multi-stream audio (conversations, songs with background tracks, movies etc.) would be to fill-in to the roles we described above and consequently can serve as real-time personal assistants. The same has been true with the chat-bot revolution we are in today, where a seemingly simple task of question-answering is unlocking capabilities as profound as code generation Ouyang et al. [2022]; Thoppilan et al. [2022]. Fundamental to all these tasks are causal audio transformations too, with looser latency constraints relative to semantic hearing, but requiring much higher-level understanding of audio signals. This chapter describes our work taking a step in that direction, tackling the problem of full-duplex voice interaction.

Despite broad interest in modeling spoken dialogue agents, most approaches are inherently "half-duplex" – restricted to turn-based interaction with responses requiring explicit prompting by the user or implicit tracking of interruption or silence events. Human dialogue, by contrast, is "full-duplex" allowing for rich synchronicity in the form of quick and dynamic turn-taking, overlapping speech, and backchanneling. Technically, the challenge of achieving full-duplex dialogue with LLMs lies in modeling synchrony as pre-trained LLMs do not have a sense of "time". To bridge this gap, we propose Synchronous LLMs for full-duplex spo-

ken dialogue modeling. We design a novel mechanism to integrate time information into Llama3-8b so that they run synchronously with the real-world clock. We also introduce a training recipe that uses 212k hours of synthetic spoken dialogue data generated from text dialogue data to create a model that generates meaningful and natural spoken dialogue, with just 2k hours of real-world spoken dialogue data. Synchronous LLMs outperform state-of-the-art in dialogue meaningfulness while maintaining naturalness. Finally, we demonstrate the model's ability to participate in full-duplex dialogue by simulating interaction between two agents trained on different datasets, while considering Internet-scale latencies of up to 240ms.

3.1 Synchronous LLMs as Full-Duplex Dialogue Agents

Existing spoken dialogue models are predominantly turn-based interfaces that are half-duplex in nature Lakhotia et al. [2021]; Zhang et al. [2023a]; Hassid et al. [2024]; Borsos et al. [2023]. To achieve a change of turn, these systems rely on either explicit user inputs or pauses at the end of a user's utterance Zhang et al. [2023a]. Human spoken dialogue, by contrast, does not rely on silence as its primary turn-taking cue Levinson and Torreira [2015a]; Nguyen et al. [2022]. Research indicates that in human conversations intra-turn pauses (pauses within a speaker's turn) are usually longer than the intervals between turns across speakers Heldner and Edlund [2010]; Brady [1968]; ten Bosch et al. [2005]. English speakers often begin their turns without waiting for pauses, using grammatical, prosodic, and pragmatic cues to seamlessly initiate their next turn while minimizing overlaps and gaps Stivers et al. [2009].

Human spoken dialogue is inherently full-duplex, allowing for seamless, bi-directional communication where both parties can simultaneously speak and listen. This mode of interaction enables immediate feedback, interruptions for clarification, and real-time adjustments in information flow Reece et al. [2023]; Levinson and Torreira [2015b]. Unlike half-duplex systems that process text or speech based on full utterances in each turn, human dialogue frequently contains verbal backchannels – short, overlapping phrases such as "yeah" or "uh-huh" – signals from the listener to the speaker that they understand and that the speaker may continue. Such synchronous dynamics allow the interaction to flow smoothly and create a rhythm absent in written text Heldner and Edlund [2010]. While humans learn turn-taking cues from infancy to minimize speech overlaps and silence duration Nguyen et al. [2021], overlapping speech as well as long silences are common in human spoken dialogue as they enrich conversations providing additional

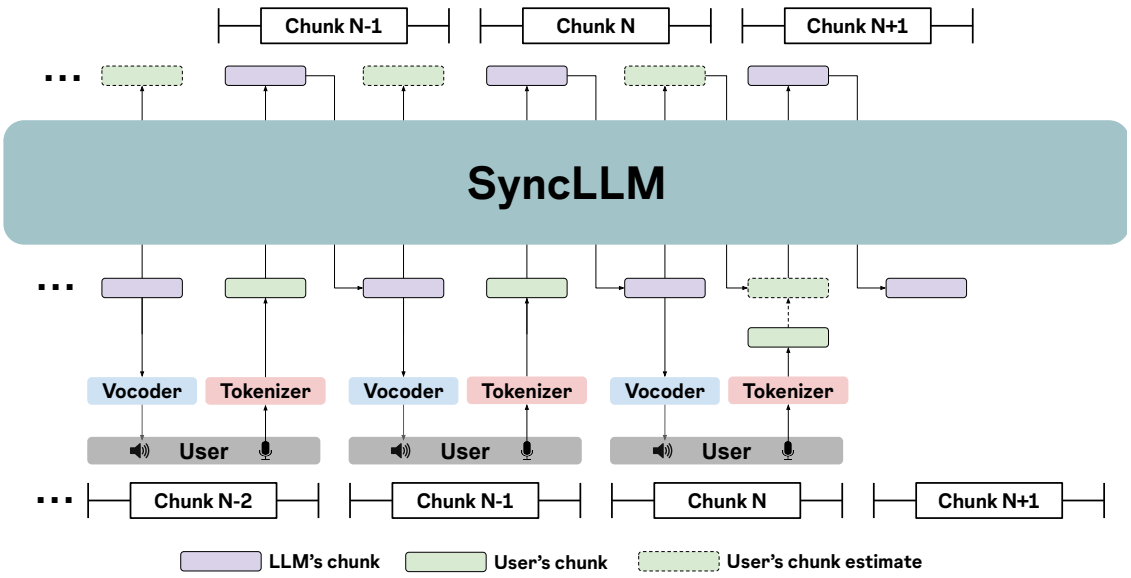


Figure 3.1: SyncLLM as a full-duplex dialogue agent. At current time step (chunk N in the figure), SyncLLM's context contains interleaved chunks of the LLM's speech until the current chunk, and the user's speech corresponding to all but the current chunk. To be in synchrony with the user, the LLM must generate its next chunk (chunk N+1) before the end of the current chunk. As a result, SyncLLM first generates an *estimated user's chunk*, which is in-turn appended to the context and used to predict its next chunk.

pragmatic cues. For example, overlapping speech and frequent backchanneling often signifies engaged listening. Similarly the length of silences can vary across cultures and is influenced by the promptness of responses Stivers et al. [2009]; Nguyen et al. [2022]. In both cases, these dynamics make conversation sound more "human."

Developing a full-duplex spoken dialog agent is challenging for four reasons: 1) Understanding and generating turn-taking cues in spoken dialogue requires the model to have a common reference clock with the real-world. However, current LLMs do not have such a sense of "time". 2) Compared to text-based chat datasets, spoken dialogue data is limited. A combination of all significant spoken dialogue datasets Cieri et al. [2004]; Godfrey et al. [1992]; Reece et al. [2023] would still result in only $\sim$3k hours of spoken dialogue data. 3) Full-duplex dialogue entails model to be always listening and should always be ready to speak, because back-channels or overlaps could occur at arbitrary points in time. This requires the model to be streaming for the duration of the dialogue. 4) Since the spoken dialogue agent might run on cloud infrastructure, it must address the fundamental latency inherent in Internet transmissions. Thus, the model may not have immediate access to the current tokens or speech generated by the user and must operate with delayed input (Fig. 3.1).

He, we make multiple contributions for developing a full-duplex dialogue agents:

• We introduce Synchronous LLMs, in short SyncLLM, for full-duplex spoken dialogue. SyncLLM achieves synchrony modeling by integrating time information into LLMs so that they can run synchronously with the real-world clock. We generate a periodic synchronization token to provide a common time frame for both sides of the dialogue. This however requires us to address duplicate tokens, caused by silence within and across utterances. Duplicate tokens can adversely affect the semantic capability of spoken dialogue model Nguyen et al. [2022]. Instead, we train our model to predict deduplicated token sequences, with timing information maintained by our periodic synchronization tokens.

• Human voice interactions rely on the ability to model the other person's response on the short-term. We can take turns with gaps as small as 200ms, while language generation latency is around 600ms Levinson and Torreira [2015b]. This implies we anticipate the next few words of what the other person would say and respond appropriately. We use this insight to predict speech units for both speakers, into the future, in chunk sizes of 160-240 ms. This ensures resiliency to Internet latencies of up to 240 ms.

• We propose a three-stage training recipe that leverages synthetic spoken dialogue generated from text dialogue data to mitigate the limited availability of real-world spoken dialogue data. Specifically, we use 212k hours of synthetic spoken dialogue data and just 2k hours of real-world spoken dialogue data to develop a model that generates meaningful spoken dialogue with naturalistic turn-taking, overlaps, and backchannels.

• With an experimental setup based on Llama3-8b at Meta [2024] and extensive user-study (n=32), we show that our method achieves +2.2-point Mean Opinion Score (MOS) improvement in dialogue content *Meaningfulness* over state-of-the-art full-duplex voice model dGSLM Nguyen et al. [2022], while maintaining turn-taking *Naturalness*. Further, our results show that our model fine-tuned on the Fisher training set Cieri et al. [2004] can generalize to the out-of-distribution Candor testset Reece et al. [2023], while preserving both dialog content meaningfulness and naturalness.

• Finally, by simulating full-duplex dialogue between two finetuned Llama3-8b models, we show how this approach can enable latency-tolerant and streaming full-duplex voice interfaces. Further, SyncLLM can perform a coherent conversation even when the user's side of the conversation is generated by a model trained with a different dataset.

3.2 Related work

Multimodal language models. The success of text language models like GPT-4 OpenAI [2023], LLAMA Touvron et al. [2023], and Mistral Jiang et al. [2023a] has inspired explorations into multimodal models. Here, we focus our discussion on speech and text modalities. Initialization from a pretrained text LLM has been shown to benefit multimodal training Hassid et al. [2023]. Recent works have proposed extending the vocabulary of text LLMs with discrete speech tokens to enable the model to handle speech inputs and outputs Rubenstein et al. [2023]. Models are trained with cross-modal knowledge from aligned speech-text data, including tasks like automatic speech recognition (ASR), text-to-speech synthesis (TTS), speech-to-text (S2T), and speech-to-speech translation (S2ST). Multitask learning with these tasks has been adopted by VioLA Wang et al. [2023b], AudioPaLM Rubenstein et al. [2023], VoxtLM Maiti et al. [2023], and SUTLM Chou et al. [2023]. SpiRit-LM Nguyen et al. [2024] interleaves speech and text tokens and trains the model with next token prediction, demonstrating both speech understanding and generation.

Spoken dialogue models. Prior work on spoken dialogue research covers various topics such as dialogue state tracking Zhang et al. [2023b], turn-taking prediction Skantze [2021]; Lin et al. [2022], and response generation Zhang et al. [2020]. Recent works leverage LLMs in dialogue systems Zhao et al. [2020]. Initialized from LLAMA, SpeechGPT Zhang et al. [2023a] is finetuned sequentially on speech-only data and multimodal instruction sets to perform spoken question answering (QA) tasks. USDM Kim et al. [2024a] continues pretraining Mistral with interleaved speech-text data to capture multimodal semantics. For dialogue finetuning, it constructs templates using both speech and transcripts of user input as instruction data. Unlike models that use speech tokens, Spectron Nachmani et al. [2023] directly manipulates spectrograms for tasks such as spoken QA and speech continuation. However, these prior works are limited to the turn-taking setting, where the dialogue model is explicitly prompted to speak in its own turn. Human spoken dialogue is more complex, involving implicit turn-taking cues and overlapping speech, such as interruptions and backchanneling Schegloff [2000].

The closest work to ours is dGSLM Lakhotia et al. [2021], which models simultaneous dialogue using a dual-tower Transformer that attends to two channels. It demonstrates superior performance than cascaded architecture which consists of automatic speech recognition (ASR), text LLM and text-to-speech (TTS). One weakness of dGSLM is its reliance on speech-only training, which does not fully utilize textual knowledge.

In contrast, our work leverages the generative intelligence of language models, equipping them with multi-modal and synchronous capabilities. Moreover, in its empirical study, dGSLM does not consider delays in real-life scenarios and assumes that the hidden states of one interlocutor are immediately accessible to the other. In contrast, we explicitly discuss how our model handles delayed responses in spoken dialogue.

3.3 SyncLLM

SyncLLM is an auto-regressive transformer decoder architecture, that natively models discrete speech units in a wall-clock synchronous fashion. SyncLLM is trained to predict interleaving chunks of speech units corresponding to both sides of the dialogue as shown in Fig. 3.1. In each time step, the model predicts speech units corresponding to a fixed duration, referred to as the model's *chunk size*, for its side of the dialogue followed by speech units corresponding to user's side of the dialogue. With this approach, the model is capable of generating two streams of speech synchronized with a real-world clock. This allows our method to model all conversational cues such as backchannels, overlaps, interruptions etc. Furthermore, since we use the same architecture as existing LLMs, our approach can leverage large scale pre-training of LLMs.

The model trained to predict interleaved chunks of token sequences can be used for full-duplex voice interaction if we could replace one of the two token streams, with that corresponding to the real-world user. In Fig. 3.1, purple boxes correspond to token sequences of the LLM's side of the conversation in each time chunk and the green boxes correspond to the user's side of the dialogue. We achieve full-duplex LLM-user voice interaction by discarding the LLM's predictions of user's response and replace it with the user's speech.

3.3.1 Latency tolerant interaction

In Fig. 3.1, consider the Nth time chunk to be current time step. We could interleave the LLM's output speech chunks until the Nth chunk, with the user's input chunks corresponding to only N-1 chunks. The reasoning here is that the user's input for the Nth chunk is not available until the end of Nth time step. To handle this intrinsic latency, similar to the way humans anticipate the next few words of what the other person taking part in the dialogue would say Levinson and Torreira [2015b], the LLM's output for the next

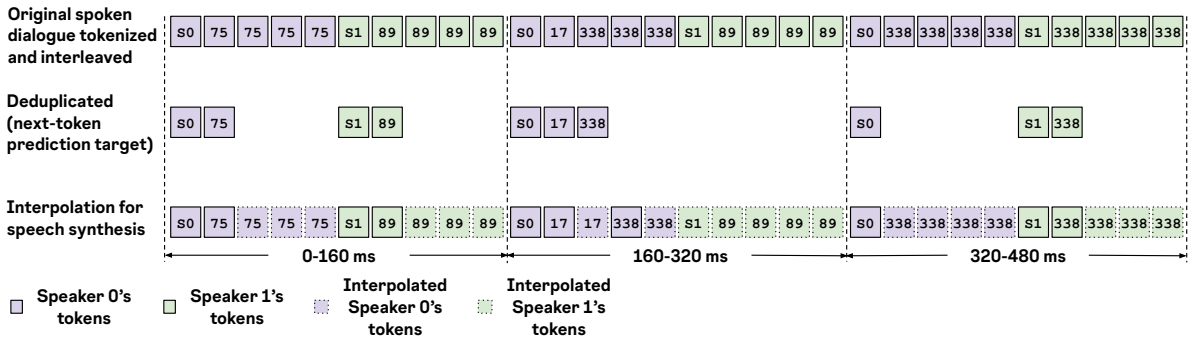


Figure 3.2: SyncLLM's token sequence format visualized with a chunk size of 160 ms. (Top row) We represent spoken dialogue as interleaved chunks of HuBERT tokens, where the chunk size determines the frequency of the synchronization token [S0]. (Middle row) We train SyncLLM to generate interleaved chunks of deduplicated HuBERT tokens along with periodic synchronization tokens. (Bottom row) We interpolate deduplicated tokens in each chunk to obtain spoken dialogue sequence in the original format.

chunk (N+1) is computed by first estimating the user's response for the Nth time chunk (depicted in the figure with green boxes with dotted border). We then append this estimated chunk to the LLM's context to generate the LLM's next chunk (N+1). For generating subsequent chunks (N+2, N+3, ...), we discard the estimated user's chunk for Nth time step and replace that with the user's real-world input, thus grounding the subsequent interaction with actual input from the user.

3.3.2 Token sequence format

Following prior works in spoken language modeling Nguyen et al. [2022, 2024], we use HuBERT Hsu et al. [2021] to represent speech. We use the tokenization parameters from Nguyen et al. [2024], with a token sampling rate of 25 Hz – resulting in one token for every 40 ms of audio – and a vocabulary size of 501. To model dialog between two speakers 0 & 1, we define two special tokens `[S0]` and `[S1]`, referred to as speaker tags, specifying the start of each speaker's token sequence, respectively. We represent dialogue as two parallel speech streams, one for each speaker, interleaved, as shown in the top row of Fig. 3.2. For each stream, we embed a periodic speaker tag, with the time period equal to chunk size of the model.

Deduplication. The fixed time period of HuBERT tokens is useful for modeling time in the full-duplex dialogue. However, raw HuBERT sequences consist of significant repeated tokens, mainly caused by silence within and across utterances. The number of repetitions of each unique token denote the duration of the acoustic unit represented by the token. The semantic content, however, can be modeled by only considering unique tokens while deduplicating the token sequence Kharitonov et al. [2022]; Nguyen et al. [2022].

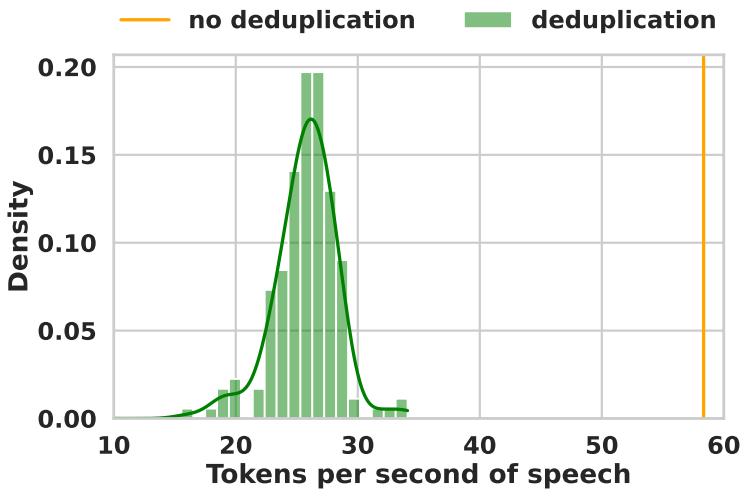


Figure 3.3: Tokens required for representing a second of speech with/without deduplication. Histogram computed over 15 hr of dialog data in the Fisher dataset Cieri et al. [2004].

Duplicate token sequences can adversely affect the semantic capability of the final spoken dialogue model Nguyen et al. [2022], because as shown in Fig. 3.3, they contain $\sim 50\%$ lower semantic content per token compared to deduplicated sequences.

So, instead, SyncLLM is trained to predict deduplicated HuBERT sequences, with coarse timing information maintained by periodically interleaved special tokens, `[S0]` and `[S1]`, as in the second row of Fig. 3.2. In the first chunk of the example in Fig. 3.2, the two speaker streams contained 4 repetitions of `[75]` and `[89]`, respectively. After deduplication, the interleaved token sequence corresponding to the first chunk would be `[S0][75][S1][89]`. In the second chunk, speaker 0 has 2 new tokens (`[17]` & `[338]`), but speaker 1 tokens are just a repetition of the last token in the previous chunk, `[89]`. So, the second chunk's token sequence would just be `[S0][17][338]`. Note that when a chunk contains no novel tokens corresponding to speaker 1, we exclude speaker 1's special token `[S1]` as well. However, this is not the case for speaker 0, as we need one of the speaker's special token to be present in all chunks to unambiguously distinguish chunks. This is shown in the third chunk of Fig. 3.2.

Interpolation. While deduplicated token sequences are beneficial for auto-regressive modeling, to generate token sequences suitable for speech synthesis, we need periodic HuBERT tokens in the original format. Since the speaker tag `[S0]` maintains the timing information, we know the number of tokens removed after deduplication within each chunk. We use this to interpolate the deduplicated token to match the expected number of token in each chunk. For example, in the first chunk of Fig. 3.2, speaker 0's stream only has one

Table 3.1: Data used for training in different stages. We convert text based data to speech using TTS.

	Stage	Source modality	Speech (hrs)
Supervised finetuning (SFT)	1	Text	193k
Dialogue	2	Text	20k
Spoken dialogue	3	Speech	1927

token after deduplication. But since chunk size in that case is 160ms, each chunk would contain 160/40 = 4 tokens. So as shown in the third row of Fig. 3.2, we repeat the deduplicated token thrice to reconstruct the chunk. If a chunk has multiple deduplicated tokens, like the second in Fig. 3.2, we repeat each token by an equal amount. We note this approach could result in an error because the original chunk may not follow this heuristic. We observed that the effect of this is imperceptible even with a chunk size of 240 ms, likely because the error in the predicted duration of each token is upper bounded by the chunk size. Further, in chunks with more novel tokens, the error would be even smaller.

3.4 Training

We use Llama3-8b at Meta [2024] as our base model and employ a three stage training procedure that uses synthetic spoken dialogue data predominantly and relatively small amount of real-world spoken dialogue data to develop a full-duplex voice agent.

Stage 1: Turn-based spoken dialogue model with synthetic speech data. Given the limited spoken dialogue data, we generate synthetic speech data from large-scale text dialogue datasets. We use supervised finetuning (SFT) datasets, as our source text-dialogue datasets. We used Bark TTS AI [2023] model to generate spoken versions of text-dialogue datasets, with its 10 speaker presets.

Since Llama3-8b is a text-only LLM, in the first stage, we aim to achieve text-speech alignment in the context of dialogues. Given a spoken question, we train the model to generate a spoken response. We expand the vocabulary of Llama3 to include 501 HuBERT tokens, in addition to the speaker tags, `[S0]` and `[S1]`. A turn-based dialog could be defined as made of turns, which in turn are made of sentences. We finetuned Llama3 with dialog sequences in the following format:

```
[S1]<sent0>[S0]<sent0><sent1>[S1]..
```

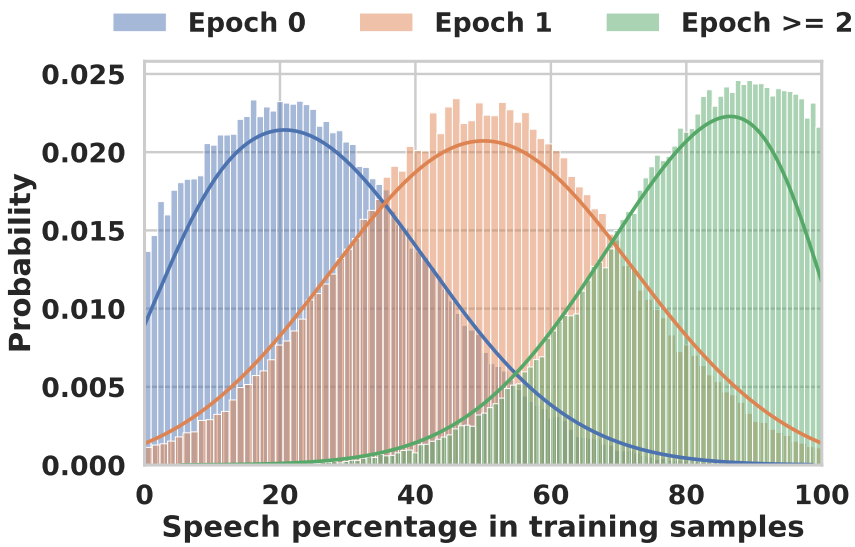


Figure 3.4: We sample speech percentages from truncated normal distribution, so we obtain samples with all possible combinations of text-speech interleaving throughout the training process, with a bias for higher speech percentages as the training progresses. This resulted in stable training when starting out with a text-only LLM.

Each sentence is randomly chosen to either be text or deduplicated speech token sequences during training. For each training sample, we sample the percentage of speech sentences in the training sequence from the truncated normal distribution (Fig. 3.4). Training only with fully speech sequences or step-wise increment of speech percentage resulted in unstable training. Sentence level text-speech interleaving not only trains the model to be capable of performing dialog, but also achieves text/speech alignment in the context of dialog.

Stage 2: Full-duplex dialogue assuming no overlaps. Turn-based spoken dialogue is special case of full-duplex dialogue with no overlaps. Based on this observation, we could treat synthetic spoken dialogue data as full-duplex spoken dialogue data where during one speaker's turn, other speaker is completely silent. In this stage, we create synthetic spoken dialogue data from text-dialogue data similarly to the previous stage with one main difference: From each turn in the dialogue, we generate a speech utterance corresponding to one speaker and silence of equal duration corresponding to the other speaker. We then tokenize the parallel speech dialog data in the format shown in the second row of Fig. 3.2. This way, we can further leverage text-dialogue data for help our model learn the token sequence format in Fig. 3.2. This stage of finetuning models timing within an utterance. The model cannot learn turn-taking cues such as back-channeling or overlaps between two speakers yet.

Table 3.2: Comparison of Pearson correlation of turn-taking event durations between generations and ground-truth continuations, given same set of prompts. SyncLLM's chunk sizes are shown in parenthesis.

Model	Fisher (in-distribution)				Candor (out-of-distribution)			
	ipu	pause	fto	Average	ipu	pause	fto	Average
dGSLM	0.48	0.41	0.10	0.33	0.30	0.02	0.09	0.14
SyncLLM-F (160 ms)	0.60	0.50	0.20	0.43	0.45	0.09	0.14	0.23
SyncLLM-F (200 ms)	0.60	0.49	0.19	0.43	0.44	0.28	0.14	0.29
SyncLLM-F (240 ms)	0.58	0.40	0.25	0.41	0.45	0.27	0.21	0.31
Prompt	0.72	0.53	0.31	0.52	0.54	0.30	0.12	0.32
Resynth-GT	0.92	0.92	0.53	0.79	0.90	0.86	0.37	0.71

For the the previous stage, most samples in SFT datasetswould contain one speaker (user of the LLM) taking a short turn and the other speaker (the LLM) giving a long response. Spoken dialogues however contain more frequent turn-taking taking with short utterances. Therefore for this stage, we use text-dialogue datasets comprising of shorter turns, equivalent to around 20k hrs of synthetic spoken dialogue.

Stage 3: Modeling with real-world spoken dialogue data. Finally, we finetune the model to learn turn-taking cues from real-world spoken dialogue data. We use the Fisher Cieri et al. [2004] dataset with 2000 hours of spoken dialogues, where each speaker's speech in a dialogue is separated into independent audio channels. We split the dataset into train, val and test split with 98:1:1 ratio, respectively. Each audio channel in the dialogue is separately tokenized and interleaved in the full-duplex dialogue format used in the previous stage. In this stage in addition to learning timing within utterances, the model learns effective turn-taking, conversational cues like accurate distribution of pauses between turn and backchanneling.

3.5 Evaluation

We evaluate SyncLLM in both continuation and interaction settings. In the continuation setting, given a spoken dialogue prompt, the model generates both sides of the dialogue. For interaction setting, we simulate interaction between two instances of SyncLLM as described in §3.3.1. We denote SyncLLM trained on Fisher in continuation setting as SyncLLM-F and use dGSLM as the continuation setting baseline. Both dGSLM and SyncLLM-F use Fisher as the only real-world spoken dialogue dataset for training. We denote SyncLLM trained on Fisher interacting with an instance trained on Fisher as SyncLLM-F-F, and SyncLLM trained on Fisher interacting with an instance trained on CANDOR Reece et al. [2023] as SyncLLM-F-C.

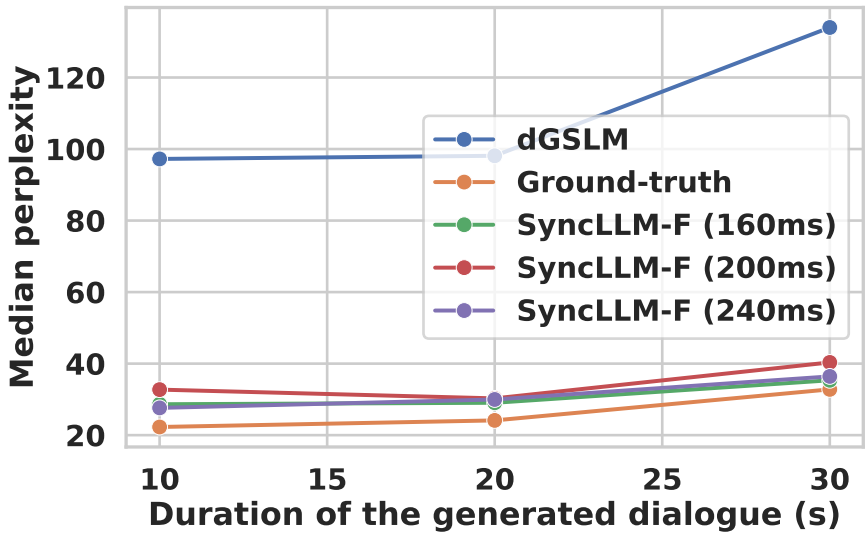


Figure 3.5: Perplexity of transcriptions of spoken dialogues generated by different models. Perplexity is measured with respect to a text dialogue model's predictions.

3.5.1 Semantic evaluation

We evaluate the semantics of SyncLLM in the text domain by converting spoken generations to text using ASR. We transcribe the generated spoken dialogues into turn-based text dialogues ignoring any overlapping speech. We then compute perplexity of transcribed dialogues generated with 10 second spoken dialogue prompts, with respect a text-only dialogue model. To account for outliers (samples with abnormally high perplexities), we consider median perplexity over the testset.

Fig. 3.5 compares the semantic quality of spoken dialogues generated by SyncLLM with different chunk sizes to the prior state-of-the-art full-duplex dGSLM model Nguyen et al. [2022] and ground-truth continuations. We find that dGSLM has a perplexity drop of $\sim$70 relative to the ground-truth, while SyncLLM only has a drop of $\sim$15. Fig. 3.6 also compares median perplexities measured with prompts sampled from Fisher and Candor test splits separately, with all models trained only on Fisher training split. Here, Candor test split is an out-of-distribution testset.

These evaluations show that our approach of using the standard auto-regressive architecture, thus leveraging vast text pre-training, results in much more semantically coherent spoken dialogue model, compared to a custom architecture proposed for speech-only training. Furthermore, our three-stage training approach leveraging large amount of synthetic spoken dialogue data generated from text dialogues, allows us to converge much faster on limited real-world dual-channel spoken dialogue data. This results in a general model

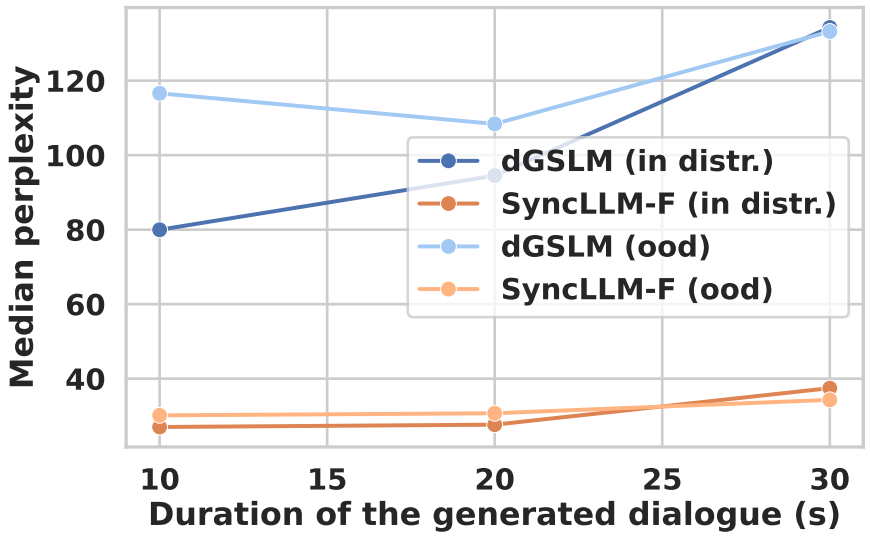


Figure 3.6: In-distribution and out-of-distribution testing.

that has superior out-of-distribution (ood) performance.

3.5.2 Naturalness evaluation

Appropriate timing of pauses, speaker transitions and overlaps are integral part of spoken-dialogue which convey essential information required for natural spoken conversation. To evaluate these aspect of our generated spoken dialogues, we consider the turn-taking events proposed in Nguyen et al. [2022] that evaluate overall naturalness of generated spoken dialogues: inter-pausal units (IPUs), pauses, and floor-transfer offset (FTO). FTO is the duration of between turn-transitions, which is a combination of overlaps and gaps – negative FTOs represent overlaps and positive FTOs represent gaps.

Similar to dGSLM's setup, we use 30s prompts sampled from the test splits and generate 90s dialogues with different model configurations. We then compute pair-wise correlation of turn-taking event durations between the dialogue generations and ground-truth continuations, given the same prompt. We first compute voice activities of each side of dialogue (generated in separate audio channels) using the `pyannote.audio` library Bredin et al. [2020]. We then measure the start and end timestamps for each turn-taking event. We measure the average duration of the turn-taking events in generated dialogues and then compute the Pearson correlation between the average durations observed in generations of different models and those in the ground-truth.

Table. 3.2 compares this correlation with in-distribution Fisher Cieri et al. [2004] test-split and out-

Table 3.3: Meaningfulness (Meaning.) and Naturalness (Nat.) (scores 1-5) mean estimates and standard errors (in parentheses), aggregated overall and for Fisher and CANDOR subsets. We use a 160ms chunk size for this study.

	Overall		**Fisher**		**CANDOR**	
Model	Meaning. ↑	Nat. ↑	Meaning. ↑	Nat. ↑	Meaning. ↑	Nat. ↑
dGSLM	1.55 (0.06)	3.95 (0.08)	1.67 (0.09)	4.21 (0.08)	1.43 (0.08)	3.70 (0.12)
SyncLLM-C	3.40 (0.07)	3.96 (0.06)	3.14 (0.10)	3.97 (0.08)	3.66 (0.08)	3.94 (0.08)
SyncLLM-F	3.74 (0.06)	3.90 (0.06)	3.82 (0.08)	3.98 (0.08)	3.67 (0.09)	3.82 (0.10)
Re-synth	3.87 (0.06)	4.03 (0.05)	4.04 (0.08)	4.14 (0.08)	3.69 (0.07)	3.91 (0.06)
GT	4.96 (0.02)	4.96 (0.02)	4.96 (0.03)	4.94 (0.04)	4.97 (0.02)	4.98 (0.02)

of-distribution Candor test-split. We observe that, generations with our models achieve better turn-taking event correlation with ground-truth continuations compared to dGSLM for both in-distribution and out-of-distribution testsets. In addition to this, we provide turn-taking event correlation with prompts and re-synthesized ground-truth continuations (Resynth-GT). Resynth-GT is obtained by re-synthesizing the tokenized ground-truth continuation. Resynth-GT does not perfectly correlate with ground-truth owing to variance in timing introduced by the tokenization process, and serves as a topline for our method.

3.5.3 Human Evaluation

We conduct an evaluation study with 32 annotators recruited via a third party vendor with the requirement that they had native-level English proficiency.

We adapt the Mean Opinion Score (MOS) protocol (a 5-pt Likert scale) ITU-T Recommendation P.808 [2018] to evaluate *Naturalness* (N-MOS) of turn-taking and *Meaningfulness* (M-MOS) of dialogue content. For both N-MOS and M-MOS, annotators are presented with the prompt- and continuation-audio. Annotators are instructed to first read the descriptions of N-MOS and M-MOS, listen to the prompt audio, then listen to the continuation audio. Finally, they are asked to provide a rating considering the quality of the continuation audio relative to the information contained in the prompt. Each annotator assigned to a given prompt / continuation pair provides a rating for both N-MOS and M-MOS.

In total, $n_{annot} = 32$ annotators provided ratings for $n_{items} = 180$ items divided evenly between the CANDOR and Fisher datasets. Each sample received a rating from $1 - Bad, ..., 5 - Excellent$ by three unique raters. We compute item-level scores by taking the median score per item. To compute system-level scores we take the mean of item scores for a given system. We compute 95% confidence intervals via

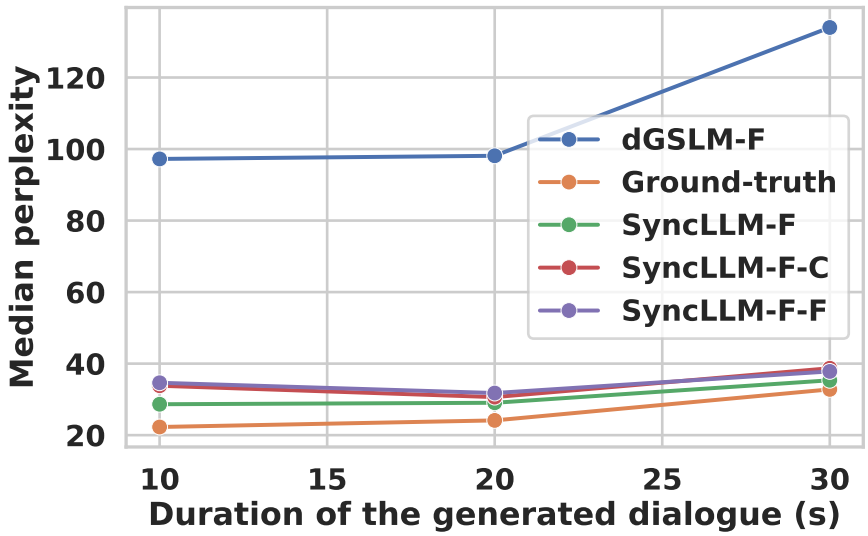


Figure 3.7: Comparison of ASR perplexity between continuation mode and interaction-mode.

bootstrapping, resampling at the item level for $n_b = 1000$ iterations.

Overall results. The two left-most columns of Table. 3.3 indicate that nearly all models are at parity in perceived *Naturalness* (N-MOS) of turn-taking, while close to re-synthesized ground-truth values. On the perceived *Meaningfulness* (M-MOS) of the dialogue content, SyncLLM-based models significantly outperform dGSLM, approaching re-synthesized ground-truth values. Resynth-GT here accounts for the tokenization process and is the topline number for the implementation of our method using the HuBERT tokenizer.

In-distribution and OOD. Table. 3.3 also highlights the difference between in-distribution (Fisher) and OOD (CANDOR) between dGSLM and Fisher-trained SyncLLM-F. While dGSLM suffers from significant degradation OOD (dropping -0.24 and -0.51 in M-MOS and N-MOS ratings), these declines are reduced in SyncLLM-F only dropping -0.15 and -0.16 moving OOD. SyncLLM trained on CANDOR dataset (SyncLLM-C) shows a decline OOD on M-MOS (-0.52), but not N-MOS (+0.03). We note that dGSLM Nguyen et al. [2022] uses speech representations fine-tuned on the Fisher dataset, while our method uses general-purpose speech representations for all domains of speech. This results in our method outperforming the baseline on the out-of-distribution Candor testset in naturalness, as judged by human evaluators in Table. 3.3.

Table 3.4: Human evaluation results for Meaningfulness (Meaning.) and Naturalness (Nat.) mean estimates and standard errors (in parentheses) across all data.

Model	Meaning. ↑	Nat. ↑
dGSLM	1.55 (0.06)	3.95 (0.08)
SyncLLM-F	3.74 (0.06)	3.90 (0.06)
SyncLLM-F-C	3.39 (0.06)	3.78 (0.06)
SyncLLM-F-F	3.47 (0.06)	3.72 (0.06)

3.5.4 Full-duplex interaction

We simulate LLM-user interaction using LLM-LLM interaction with one-chunk latency. We evaluate our model trained with different chunk sizes, thus simulating different latencies. We also train a version of SyncLLM with Candor training split in the third training stage, and simulate its interaction with the original model trained with only Fisher.

In Fig. 3.7, we compare median perplexities obtained with prompts sampled from Fisher and Candor test splits. We also show the perplexity of ground-truth and samples generated in the dialog continuation setting for reference. We find that SyncLLM in the LLM-LLM interaction setting is able to closely match the performance of the continuation setting, and perform significantly better than dGSLM in continuation setting. Furthermore, we find that interaction between instances of SyncLLM trained with Fisher and Candor datasets, respectively is are almost the same signifying that SyncLLM can perform a coherent conversation even when user's side of the conversation is generated by a model trained with a different dataset.

Human evaluation. Table. 3.4 shows ratings for dGSLM, the Fisher-trained continuation model, and LLM-LLM interactions. Results corroborate findings in §3.5.4 – LLM-LLM interactions outperform dGSLM on M-MOS, but are slightly worse compared to the single model continuation setting.

3.6 Robustness to interfering speech

While the previous method demonstrated full-duplex interaction, there are few missing components in making it deployable. First, the model still has to learn with a few thousand hours of real-world spoken dialogue data, because the dialogues have to separated into different channels. But there is so much conversation data available in the world, a more scalable approach that could leverage all of that data. The current model trained on speaker invariant speech representations fail with interfering speech, more general sound repre-

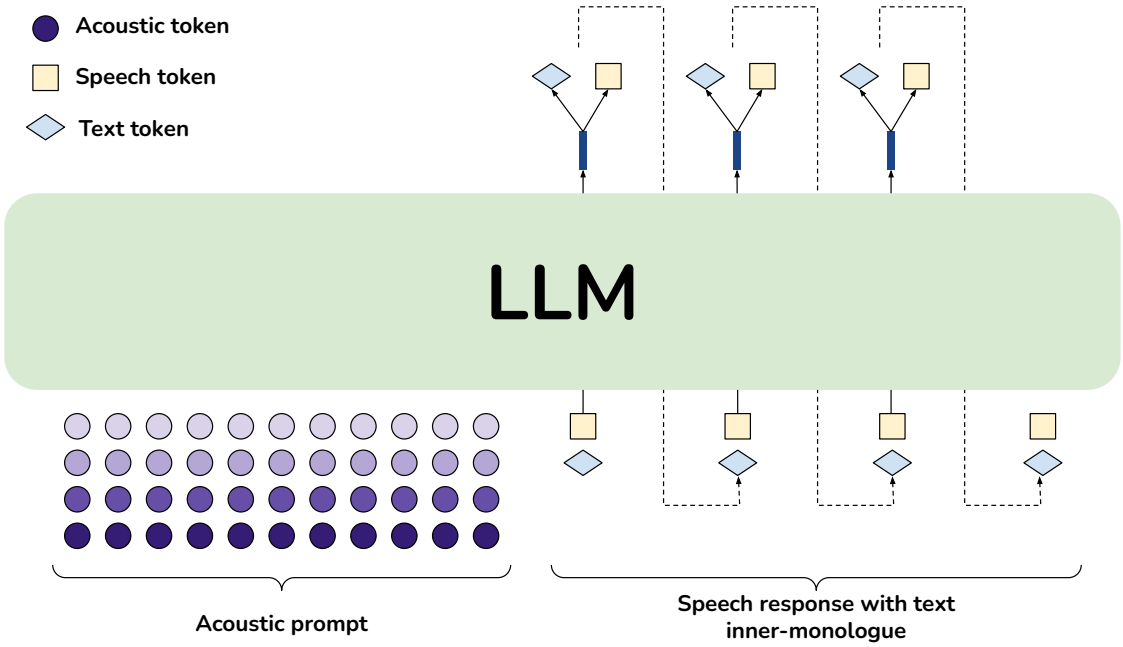


Figure 3.8: Spoken dialogue model trained to understand acoustic tokens (shown as multiple streams of purple circles corresponding to multiple codebooks) and generate text/speech representations. When generating speech response, it is beneficial to concurrently generate corresponding text tokens. Training the model to generate speech and text streams concurrently results in stronger alignment between speech and text modalities.

sentations might make it more robust to noise.

Humans understand all sounds and generate speech. If a model can understand the audio as it is, not just its speech component, then may be it'll allow us to train with all publicly available audio data on the internet. And more importantly, a model that understand all sounds is inherently more robust to noise because, it know what to focus on and what to ignore. At each timestep speaker invariant speech representations take about 500 states. In contrast, best audio tokenizers Défossez et al. [2022]; Kumar et al. [2023] can represent audio with 16 streams of token, where token in each stream can take 1024 states. This implies that each timestep can take a total of 2^{160} states. Clearly, flattening all states to a single stream is not tractable, so this forces us to model each stream separately. In this section, we investigate dialogue modeling with raw acoustic representations.

In real-world scenarios where spoken language model are deployed, we would often have background noises when the user is speaking. Prevalent speech representations such as HuBERTHsu et al. [2021] are robust to background noises if the noise is uncorrelated to speech. However, due to their speaker invariance, they amplify spurious speech noise in the background. Here we propose that a model that can understand

general audio representations Défossez et al. [2022]; Kumar et al. [2023] would be inherently capable of who the user is based on their voice characteristic and a result, would be more robust to in-domain (speech) noise, greatly enhancing the usability of spoken language models.

3.6.1 Spoken dialogue models with acoustic representations

Towards making spoken dialogue models robust to correlated background noise, we could modify the stage 1 of SyncLLM's training by providing acoustic representations in the prompt, while keeping the response is text or semantic speech representations Liu et al. [2023]. The would allow the model to understand speech input from the user, without the usual loss of intonation and voice characteristics resulting from speaker-invariant speech representations. To achieve this, similar to SyncLLM's approach, we expand the LLM's vocabulary to incorporate acoustic tokens, in addition to speech tokens.

We employ synthetic spoken dialogue data described in §3.4 to train a noise robust turn-based dialog model. As shown in Fig. 3.8, spoken prompt for the model is provided as multiple streams of acoustic tokens. At each timestep, embeddings corresponding to each of the acoustic streams are summed up before providing them to the transformer blocks. The model is then trained to predict aligned text and speech token streams, with multiple projections heads. Concurrently generating text and speech streams has been to be beneficial for strong text & speech modality alignment Défossez et al. [2024]. In order for the LLM to learn to focus on the target while ignoring noise, we mix the clean spoken prompt with -10dB to -5dB of correlated (speech) noise as well as uncorrelated noise. We used prompts from different speakers in the same dataset as the source of correlated noise to make sure there is no acoustic property difference between noise and spoken prompts. We use 10s clips from Gemmeke et al. [2017a], as the source for uncorrelated background noises.

3.6.2 Dialogue evaluation using an LLM-as-a-judge

For evaluating the effectiveness of provided spoken prompt as acoustic representations, we train the following spoken dialogue models with different input and output modality combinations:

- `speech2speech`: Given a spoken dialogue prompt as DinoSR units Liu et al. [2023] (semantic speech representations), the dialogue model is trained to generate a spoken response as DinoSR units.

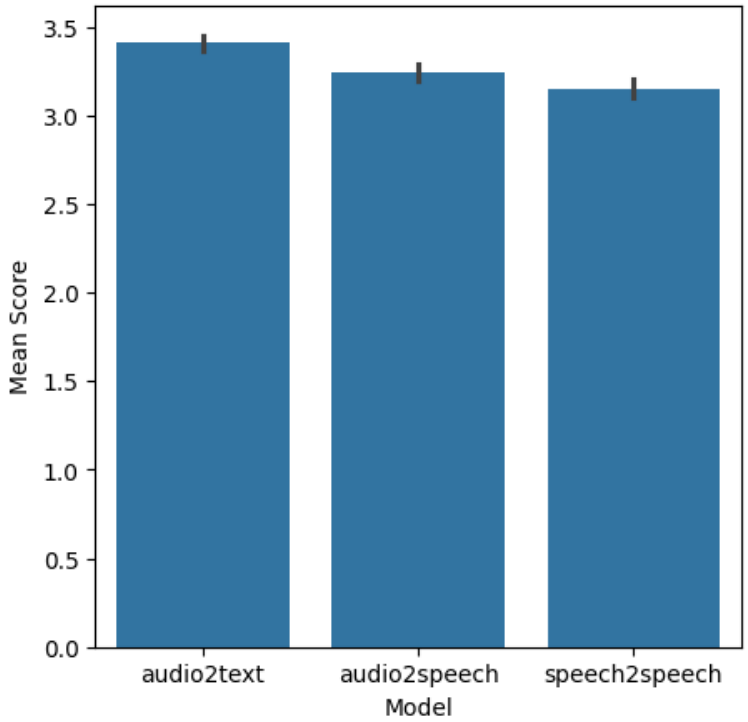


Figure 3.9: Performance of dialogue models with semantic speech unit prompts (DinoSR) vs acoustic unit prompts. Contrary to the prevalent opinion that semantic speech units are necessary for good performance, we observe that in realistic noisy situations models that are able to understand raw acoustic representations are able to perform better.

Concurrently, the model is also trained to predict aligned text stream corresponding to the spoken response.

- `audio2speech`: This model is trained to generate a spoke response in the same format as above, but the spken prompt is provided as acoustic representations. Here, we use DAC units Kumar et al. [2023] as the acoustic representations.

- `audio2text`: For a given audio prompt, this model is trained to provide a text response. This model serves as a top line for the dialogue performance with acoustic unit prompts.

In addition to these, we train a text-only dialogue model, `audio2text`, using the same set of supervised fine-tuning Touvron et al. [2023] datasets that were originally used for generating out synthetic datasets. We use Prometheus Kim et al. [2024b], LLM-as-a-judge framework to evaluate the quality of responses of different models. We use a common set of noise augmented prompts represented in text, DinoSR units and DAC units as our test set. The responses are then evaluated with Prometheus judge models to ob-

tain an score between 1-5 for each response. We used `audio2text` as the reference responses. The speech and acoustic responses are transcribed using WhisperX Bain et al. [2023] library to obtain text equivalent of the responses for LLM-based evaluation. We compare the performance of different model in Fig. 3.9. We observe that in correlated noisy settings, `audio2speech` outperforms the `speech2speech` model while also opening the door for new capabilities such as general audio and music understanding.

Chapter 4

Conclusion and Future Work

This thesis first presented methods for several capabilities we collectively refer to as *semantic hearing*, that allow us to essentially program acoustic scenes around us. Such methods provide users with enhanced and even super human hearing capabilities. This shows a glimpse into the bigger vision of the augmenting human auditory perception with AI. And in fact, we believe that in the next decade, AI will become more and more ubiquitous on hearables to enhance and augment human perception.

We then approached the problem of full-duplex voice interaction between an agent and user as yet another streaming waveform to waveform transformation. We show that the existing auto-regressive language models are compatible with this framing. So, we leveraged a pretrained language model to develop a more meaningful full-duplex voice model, compared to the state-of-the-art. Unlike previous problems where the input latency is virtually negligible, there could be internet-scale latency in receiving the input stream. In this work, we also propose techniques that are robust to input latency. Finally, we discuss the practical issues due to interfering speech during the deployment of spoke dialogue models, and show that spoken dialogue models trained with acoustic representations are more robust.

The recent emergence of sophisticated reasoning models Huang and Chang [2023] suggests that semantic consistency and naturalness may become insufficient for full-duplex agents. Users will likely expect these agents to possess comparable reasoning capabilities to state-of-the-art models. A significant performance gap would force users to choose between human-like conversational agents and high-performing models with potentially inconvenient interfaces. Bridging this gap will be crucial for the widespread adoption

of full-duplex agents. Drawing inspiration from human customer care operators who consult experienced managers, a robust full-duplex agent could be designed to query a more intelligent model when faced with complex user requests. Such an architecture would enable the agent to not only engage users in human-like conversations but also leverage the power of a reasoning model to address intricate queries effectively.

The power of large-scale self-supervised learning, well-established in text modeling, remains largely untapped for audio, despite the vast amounts of available digitized audio data. Recent work such as Moshi Défossez et al. [2024] represents progress, scaling audio pre-training to 2 million hours. However, at a token frequency of 12.5 Hz, this yields less than 0.1 trillion tokens, dwarfed by the $\sim$15 trillion tokens used in contemporary text pre-training (a 150x difference). This significant gap presents a compelling opportunity: by matching the scale of text pre-training in the audio domain, future large models could unlock seamless transfer learning between textual and auditory modalities, leading to more versatile and powerful AI systems.

Bibliography

2023. Apple AirPods. https://www.apple.com/airpods/.

2023. Audio latency meter for iOS. https://onyx3.com/LatencyMeter/.

2023. Customize transparency mode for AirPods Pro. https://support.apple.com/guide/airpods/customize-transparency-mode-dev966f5f818/web.

2023. Simplefreefieldhrir. https://www.sofaconventions.org/mediawiki/index.php.

Suno AI. 2023. Bark tts. https://github.com/suno-ai/bark.

V.R. Algazi, R.O. Duda, D.M. Thompson, and C. Avendano. 2001. The cipic hrtf database.

Taichi Asami, Ryo Masumura, Yoshikazu Yamaguchi, Hirokazu Masataki, and Yushi Aono. 2017. Domain adaptation of dnn acoustic models using knowledge distillation.

Alexei Baevski, Henry Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations.

Junjie Bai, Fang Lu, Ke Zhang, et al. 2019. ONNX: Open neural network exchange. https://github.com/onnx/onnx.

Max Bain, Jaesung Huh, Tengda Han, and Andrew Zisserman. 2023. Whisperx: Time-accurate speech transcription of long-form audio. *INTERSPEECH 2023*.

Aaron Bangor, Philip T. Kortum, and James T. Miller. 2008. An empirical evaluation of the system usability scale. *International Journal of Human–Computer Interaction*, 24(6):574–594.

Loïc Barrault, Yu-An Chung, Mariano Cora Meglioli, David Dale, Ning Dong, Paul-Ambroise Duquenne, Hady Elsahar, Hongyu Gong, Kevin Heffernan, John Hoffman, et al. 2023. Seamlessm4t-massively multilingual & multimodal machine translation. *arXiv preprint arXiv:2308.11596*.

Zalán Borsos, Raphaël Marinier, Damien Vincent, Eugene Kharitonov, Olivier Pietquin, Matt Sharifi, Dominik Roblek, Olivier Teboul, David Grangier, Marco Tagliasacchi, and Neil Zeghidour. 2023. Audiolm: a language modeling approach to audio generation.

Louis ten Bosch, Nelleke Oostdijk, and Lou Boves. 2005. On temporal aspects of turn taking in conversational dialogues. *Speech Commun.*, 47:80–86.

Paul T. Brady. 1968. A statistical analysis of on-off patterns in 16 conversations. *Bell System Technical Journal*, 47:73–91.

Hervé Bredin, Ruiqing Yin, Juan Manuel Coria, Gregory Gelly, Pavel Korshunov, Marvin Lavechin, Diego Fustes, Hadrien Titeux, Wassim Bouaziz, and Marie-Philippe Gill. 2020. pyannote.audio: neural building blocks for speaker diarization. In *ICASSP 2020, IEEE International Conference on Acoustics, Speech, and Signal Processing*, Barcelona, Spain.

John Brooke. 1995. Sus: A quick and dirty usability scale. *Usability Eval. Ind.*, 189.

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, Harsha Nori, Hamid Palangi, Marco Tulio Ribeiro, and Yi Zhang. 2023. Sparks of artificial general intelligence: Early experiments with gpt-4.

Nam Bui, Nhat Pham, Jessica Jacqueline Barnitz, Zhanan Zou, Phuc Nguyen, Hoang Truong, Taeho Kim, Nicholas Farrow, Anh Nguyen, Jianliang Xiao, Robin Deterding, Thang Dinh, and Tam Vu. 2021. Ebp: An ear-worn device for frequent and comfortable blood pressure monitoring. *Commun. ACM*.

Justin Chan, Nada Ali, Ali Najafi, Anna Meehan, Lisa Mancl, Emily Gallagher, Randall Bly, and Shyamnath Gollakota. 2022. An off-the-shelf otoacoustic-emission probe for hearing screening via a smartphone. *Nature Biomedical Engineering*, 6:1–11.

Justin Chan, Sharat Raju, Rajalakshmi Nandakumar, Randall Bly, and Shyamnath Gollakota. 2019. Detecting middle ear fluid using smartphones. *Science Translational Medicine*, 11:eaav1102.

Ishan Chatterjee, Maruchi Kim, Vivek Jayaram, Shyamnath Gollakota, Ira Kemelmacher, Shwetak Patel, and Steven M Seitz. 2022. Clearbuds: wireless binaural earbuds for learning-based speech enhancement.

Ke Chen, Xingjian Du, Bilei Zhu, Zejun Ma, Taylor Berg-Kirkpatrick, and Shlomo Dubnov. 2022. Zero-shot audio source separation through query-based learning from weakly-labeled data. In *AAAI*, volume 36.

Xie Chen, Yu Wu, Zhenghao Wang, Shujie Liu, and Jinyu Li. 2021. Developing real-time streaming transformer transducer for speech recognition on large-scale dataset. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5904–5908.

Zhuo Chen, Naoyuki Kanda, Jian Wu, Yu Wu, Xiaofei Wang, Takuya Yoshioka, Jinyu Li, Sunit Sivasankaran, and Sefik Emre Eskimez. 2023. Speech separation with large-scale self-supervised learning. In *ICASSP 2023-2023 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 1–5. IEEE.

Ju-Chieh Chou, Chung-Ming Chien, Wei-Ning Hsu, Karen Livescu, Arun Babu, Alexis Conneau, Alexei Baevski, and Michael Auli. 2023. Toward joint language modeling for speech units and text.

Christopher Cieri, David Miller, and Kevin Walker. 2004. The fisher corpus: a resource for the next generations of speech-to-text. In *International Conference on Language Resources and Evaluation*.

Alexandre Défossez, Laurent Mazaré, Manu Orsini, Amélie Royer, Patrick Pérez, Hervé Jégou, Edouard Grave, and Neil Zeghidour. 2024. Moshi: a speech-text foundation model for real-time dialogue. Technical report.

Marc Delcroix, Jorge Bennasar Vázquez, Tsubasa Ochiai, Keisuke Kinoshita, and Shoko Araki. 2021. Few-shot learning of new sound classes for target sound extraction. *arXiv preprint arXiv:2106.07144*.

Marc Delcroix, Jorge Bennasar Vázquez, Tsubasa Ochiai, Keisuke Kinoshita, Yasunori Ohishi, and Shoko Araki. 2022a. Soundbeam: Target sound extraction conditioned on sound-class labels and enrollment clues for increased performance and continuous learning. In *arXiv*.

Marc Delcroix, Jorge Bennasar Vázquez, Tsubasa Ochiai, Keisuke Kinoshita, Yasunori Ohishi, and Shoko Araki. 2022b. Soundbeam: Target sound extraction conditioned on sound-class labels and enrollment clues for increased performance and continuous learning.

Florian Denk, Henning Schepker, Simon Doclo, and Birger Kollmeier. 2020. Acoustic transparency in hearables—technical evaluation. *Journal of the Audio Engineering Society*, 68(7/8):508–521.

Alexandre Défossez, Jade Copet, Gabriel Synnaeve, and Yossi Adi. 2022. High fidelity neural audio compression. *arXiv preprint arXiv:2210.13438*.

Alexandre Défossez, Nicolas Usunier, Léon Bottou, and Francis Bach. 2021. Music source separation in the waveform domain.

Ariel Ephrat, Inbar Mosseri, Oran Lang, Tali Dekel, Kevin Wilson, Avinatan Hassidim, William T Freeman, and Michael Rubinstein. 2018. Looking to listen at the cocktail party: A speaker-independent audio-visual model for speech separation. *arXiv preprint arXiv:1804.03619*.

Sefik Emre Eskimez, Takuya Yoshioka, Huaming Wang, Xiaofei Wang, Zhuo Chen, and Xuedong Huang. 2022. Personalized speech enhancement: New models and comprehensive evaluation. In *IEEE ICASSP*.

Eduardo Fonseca, Manoj Plakal, Frederic Font, Daniel P. W. Ellis, Xavier Favory, Jordi Pons, and Xavier Serra. 2018. General-purpose tagging of freesound audio with audioset labels: Task description, dataset, and baseline. In *DCASE2018*.

Ruohan Gao and Kristen Grauman. 2019. Co-separating sounds of visual objects. In *IEEE /CVF ICCV*.

Jort F. Gemmeke, Daniel P. W. Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R. Channing Moore, Manoj Plakal, and Marvin Ritter. 2017a. Audio set: An ontology and human-labeled dataset for audio events. In *IEEE ICASSP*.

Jort F Gemmeke, Daniel PW Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R Channing Moore, Manoj Plakal, and Marvin Ritter. 2017b. Audio set: An ontology and human-labeled dataset for audio events. In *ICASSP*. IEEE.

Beat Gfeller, Dominik Roblek, and Marco Tagliasacchi. 2021. One-shot conditional audio filtering of arbitrary sounds. In *ICASSP*. IEEE.

Ritwik Giri, Shrikant Venkataramani, Jean-Marc Valin, Umut Isik, and Arvindh Krishnaswamy. 2021. Personalized percepnet: Real-time, low-complexity target voice separation and enhancement. In *arXiv*.

J.J. Godfrey, E.C. Holliman, and J. McDaniel. 1992. Switchboard: telephone speech corpus for research and development. In *[Proceedings] ICASSP-92: 1992 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 517–520 vol.1.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. 2006. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376.

Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and Ruoming Pang. 2020. Conformer: Convolution-augmented transformer for speech recognition.

Rishabh Gupta, Rishabh Ranjan, Jianjun He, Woon-Seng Gan, and Santi Peksi. 2020. Acoustic transparency in hearables for augmented reality audio: Hear-through techniques review and challenges. In *Audio Engineering Society Conference on Audio for Virtual and Augmented Reality*.

Cong Han, Yi Luo, and Nima Mesgarani. 2020. Real-time binaural speech separation with preserved spatial cues. In *arXiv*.

Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, and Andrew Y. Ng. 2014. Deep speech: Scaling up end-to-end speech recognition.

Michael Hassid, Tal Remez, Tu Anh Nguyen, Itai Gat, Alexis Conneau, Felix Kreuk, Jade Copet, Alexandre Défossez, Gabriel Synnaeve, Emmanuel Dupoux, Roy Schwartz, and Yossi Adi. 2023. Textually pretrained speech language models. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Michael Hassid, Tal Remez, Tu Anh Nguyen, Itai Gat, Alexis Conneau, Felix Kreuk, Jade Copet, Alexandre Defossez, Gabriel Synnaeve, Emmanuel Dupoux, Roy Schwartz, and Yossi Adi. 2024. Textually pretrained speech language models.

Liwei He, Elizabeth Sanocki, Anoop Gupta, and Jonathan Grudin. 1999. Auto-summarization of audio-video presentations. In *Proceedings of the Seventh ACM International Conference on Multimedia (Part 1)*, MULTIMEDIA '99, page 489–498, New York, NY, USA. Association for Computing Machinery.

Headphonesty. 2022. The fascinating history of noise-cancelling headphones. `https://www.headphonesty.com/2020/10/history-of-noise-cancelling-headphones/`.

Mattias Heldner and Jens Edlund. 2010. Pauses, gaps and overlaps in conversations. *Journal of Phonetics*, 38(4):555–568.

John R Hershey, Zhuo Chen, Jonathan Le Roux, and Shinji Watanabe. 2016. Deep clustering: Discriminative embeddings for segmentation and separation. In *2016 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 31–35. IEEE.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network.

Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. 2021. Hubert: Self-supervised speech representation learning by masked prediction of hidden units.

Yanxin Hu, Yun Liu, Shubo Lv, Mengtao Xing, Shimin Zhang, Yihui Fu, Jian Wu, Bihong Zhang, and Lei Xie. 2020. Dccrn: Deep complex convolution recurrent network for phase-aware speech enhancement.

Jie Huang and Kevin Chen-Chuan Chang. 2023. Towards reasoning in large language models: A survey. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 1049–1065, Toronto, Canada. Association for Computational Linguistics.

Po-Sen Huang, Minje Kim, Mark Hasegawa-Johnson, and Paris Smaragdis. 2014. Deep learning for monaural speech separation. In *ICASSP*. IEEE.

IoSR-Surrey. 2016. Iosr-surrey/realroombrirs: Binaural impulse responses captured in real rooms. https://github.com/IoSR-Surrey/RealRoomBRIRs.

IoSR-Surrey. 2023. Simulated room impulse responses. https://iosr.uk/software/index.php.

Yusuf Isik, Jonathan Le Roux, Zhuo Chen, Shinji Watanabe, and John R Hershey. 2016. Single-channel multi-speaker separation using deep clustering. *arXiv preprint arXiv:1607.02173*.

ITU-T Recommendation P.808. 2018. Subjective evaluation of speech quality with a crowdsourcing approach.

Mohammad Javad Jafari, Reza Khosrowabadi, Soheila Khodakarim, and Farough Mohammadian. 2019. The effect of noise exposure on cognitive performance and brain activity patterns. *Open access Macedonian journal of medical sciences*, 7(17):2924.

Dhruv Jain, Kelly Mack, Akli Amrous, Matt Wright, Steven Goodman, Leah Findlater, and Jon E. Froehlich. 2020a. Homesound: An iterative field deployment of an in-home sound awareness system for deaf or hard of hearing users. In *ACM CHI*.

Dhruv Jain, Hung Ngo, Pratyush Patel, Steven Goodman, Leah Findlater, and Jon Froehlich. 2020b. Soundwatch: Exploring smartwatch-based deep learning approaches to support sound awareness for deaf and hard of hearing users. In *ACM SIGACCESS ASSETS*.

Teerapat Jenrungrot, Vivek Jayaram, Steve Seitz, and Ira Kemelmacher-Shlizerman. 2020. The cone of silence: Speech separation by localization.

Ye Jia, Ron J. Weiss, Fadi Biadsy, Wolfgang Macherey, Melvin Johnson, Zhifeng Chen, and Yonghui Wu. 2019. Direct speech-to-speech translation with a sequence-to-sequence model. In *Interspeech*.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023a. Mistral 7b. *CoRR*, abs/2310.06825.

Lavender Jiang, Xujin Liu, Nima Nejatian, Mustafa Nasir-Moin, Duo Wang, Anas Abidin, Kevin Eaton, Howard Riina, Ilya Laufer, Paawan Punjabi, Madeline Miceli, Nora Kim, Cordelia Orillac, Zane Schnurman, Christopher Livia, Hannah Weiss, David Kurland, Sean Neifert, Yosef Dastagirzada, and Eric Oermann. 2023b. Health system-scale language models are all-purpose prediction engines. *Nature*, 619:1–6.

Wenyu Jin, Tim Schoof, and Henning Schepker. 2022. Individualized hear-through for acoustic transparency using PCA-based sound pressure estimation at the eardrum. In *ICASSP*.

Ilya Kavalerov, Scott Wisdom, Hakan Erdogan, Brian Patton, Kevin Wilson, Jonathan Le Roux, and John R Hershey. 2019. Universal sound separation. In *2019 WASPAA*. IEEE.

Eugene Kharitonov, Ann Lee, Adam Polyak, Yossi Adi, Jade Copet, Kushal Lakhotia, Tu-Anh Nguyen, Morgane Rivière, Abdelrahman Mohamed, Emmanuel Dupoux, and Wei-Ning Hsu. 2022. Text-free prosody-aware generative spoken language modeling.

Kevin Kilgour, Beat Gfeller, Qingqing Huang, Aren Jansen, Scott Wisdom, and Marco Tagliasacchi. 2022. Text-driven separation of arbitrary sounds. In *arXiv*.

Heeseung Kim, Soonshin Seo, Kyeongseok Jeong, Ohsung Kwon, Jungwhan Kim, Jaehong Lee, Eunwoo Song, Myungwoo Oh, Sungroh Yoon, and Kang Min Yoo. 2024a. Unified speech-text pretraining for spoken dialog modeling. *CoRR*, abs/2402.05706.

Seungone Kim, Juyoung Suk, Shayne Longpre, Bill Yuchen Lin, Jamin Shin, Sean Welleck, Graham Neubig, Moontae Lee, Kyungjae Lee, and Minjoon Seo. 2024b. Prometheus 2: An open source language model specialized in evaluating other language models.

Nithin Rao Koluguri, Taejin Park, and Boris Ginsburg. 2021. Titanet: Neural model for speaker representation with 1d depth-wise separable convolutions and global context.

Qiuqiang Kong, Yuxuan Wang, Xuchen Song, Yin Cao, Wenwu Wang, and Mark D Plumbley. 2020. Source separation with weakly labelled data: An approach to computational auditory scene analysis. In *ICASSP*. IEEE.

Rithesh Kumar, Prem Seetharaman, Alejandro Luebs, Ishaan Kumar, and Kundan Kumar. 2023. High-fidelity audio compression with improved rvqgan.

Kushal Lakhotia, Evgeny Kharitonov, Wei-Ning Hsu, Yossi Adi, Adam Polyak, Benjamin Bolte, Tu-Anh Nguyen, Jade Copet, Alexei Baevski, Adelrahman Mohamed, and Emmanuel Dupoux. 2021. Generative spoken language modeling from raw audio.

Gierad Laput, Karan Ahuja, Mayank Goel, and Chris Harrison. 2018. Ubicoustics: Plug-and-play acoustic activity recognition. In *ACM UIST*.

Matthew Le, Apoorv Vyas, Bowen Shi, Brian Karrer, Leda Sari, Rashel Moritz, Mary Williamson, Vimal Manohar, Yossi Adi, Jay Mahadeokar, and Wei-Ning Hsu. 2023. Voicebox: Text-guided multilingual universal speech generation at scale.

Stephen C. Levinson and Francisco Torreira. 2015a. Timing in turn-taking and its implications for processing models of language. *Frontiers in Psychology*, 6.

Stephen C Levinson and Francisco Torreira. 2015b. Timing in turn-taking and its implications for processing models of language. *Frontiers in psychology*, 6:731.

Luca Della Libera, Cem Subakan, Mirco Ravanelli, Samuele Cornell, Frédéric Lepoutre, and François Grondin. 2024. Resource-efficient separation transformer.

Ting-En Lin, Yuchuan Wu, Fei Huang, Luo Si, Jian Sun, and Yongbin Li. 2022. Duplex conversation: Towards human-like interaction in spoken dialogue systems. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '22. ACM.

Alexander H. Liu, Heng-Jui Chang, Michael Auli, Wei-Ning Hsu, and James R. Glass. 2023. Dinosr: Self-distillation and online clustering for self-supervised speech representation learning. *ArXiv*, abs/2305.10005.

Xubo Liu, Haohe Liu, Qiuqiang Kong, Xinhao Mei, Jinzheng Zhao, Qiushi Huang, Mark D Plumbley, and Wenwu Wang. 2022. Separate what you describe: Language-queried audio source separation. In *arXiv*.

Hong Lu, Wei Pan, Nicholas D. Lane, Tanzeem Choudhury, and Andrew T. Campbell. 2009. Soundsense: Scalable sound sensing for people-centric applications on mobile phones. In *ACM MobiSys*.

Yen-Ju Lu, Xuankai Chang, Chenda Li, Wangyou Zhang, Samuele Cornell, Zhaoheng Ni, Yoshiki Masuyama, Brian Yan, Robin Scheibler, Zhongqiu Wang, Yu Tsao, Yanmin Qian, and Shinji Watanabe. 2022. Espnet-se++: Speech enhancement for robust speech recognition, translation, and understanding.

Jian Luo, Jianzong Wang, Ning Cheng, Edward Xiao, Xulong Zhang, and Jing Xiao. 2022. Tiny-sepformer: A tiny time-domain transformer network for speech separation. In *arXiv*.

Yi Luo, Zhuo Chen, and Takuya Yoshioka. 2020. Dual-path RNN: efficient long sequence modeling for time-domain single-channel speech separation. In *ICASSP*. IEEE.

Yi Luo and Nima Mesgarani. 2019a. Conv-tasnet: Surpassing ideal time–frequency magnitude masking for speech separation. *IEEE/ACM transactions on audio, speech, and language processing*.

Yi Luo and Nima Mesgarani. 2019b. Conv-tasnet: Surpassing ideal time–frequency magnitude masking for speech separation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 27(8):1256–1266.

Dong Ma, Andrea Ferlini, and Cecilia Mascolo. 2021. OESense: Employing occlusion effect for in-ear human sensing. In *MobiSys*.

Soumi Maiti, Yifan Peng, Shukjae Choi, Jee-weon Jung, Xuankai Chang, and Shinji Watanabe. 2023. Voxtlm: unified decoder-only models for consolidating speech recognition/synthesis and speech/text continuation tasks. *CoRR*, abs/2309.07937.

Tobias May, Steven van de Par, and Armin Kohlrausch. 2011. A probabilistic model for robust localization based on a binaural auditory front-end. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(1):1–13.

Emma McDonnell, Soo Hyun Moon, Lucy Jiang, Steven Goodman, Raja Kushalnaga, Jon Froehlich, and Leah Findlater. 2023. "easier or harder, depending on who the hearing person is": Codesigning videoconferencing tools for small groups with mixed hearing status. In *ACM CHI*.

Annamaria Mesaros, Aleksandr Diment, Benjamin Elizalde, Toni Heittola, Emmanuel Vincent, Bhiksha Raj, and Tuomas Virtanen. 2019. Sound event detection in the dcase 2017 challenge. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 27(6):992–1006.

Annamaria Mesaros, Toni Heittola, Emmanouil Benetos, Peter Foster, Mathieu Lagrange, Tuomas Virtanen, and Mark D. Plumbley. 2018a. Detection and classification of acoustic scenes and events: Outcome of the dcase 2016 challenge. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(2):379–393.

Annamaria Mesaros, Toni Heittola, and Tuomas Virtanen. 2018b. A multi-device dataset for urban acoustic scene classification. In *DCASE*.

AT at Meta. 2024. Meta llama 3. `https://github.com/meta-llama/llama3`.

Vimal Mollyn, Karan Ahuja, Dhruv Verma, Chris Harrison, and Mayank Goel. 2022. Samosa: Sensing activities with motion and subsampled audio. *IMWUT*.

Eliya Nachmani, Alon Levkovitch, Roy Hirsch, Julian Salazar, Chulayuth Asawaroengchai, Soroosh Mariooryad, Ehud Rivlin, RJ Skerry-Ryan, and Michelle Tadmor Ramanovich. 2023. Spoken question answering and speech continuation using spectrogram-powered llm. In *The Twelfth International Conference on Learning Representations*.

Tu Anh Nguyen, Eugene Kharitonov, Jade Copet, Yossi Adi, Wei-Ning Hsu, Ali Elkahky, Paden Tomasello, Robin Algayres, Benoit Sagot, Abdelrahman Mohamed, and Emmanuel Dupoux. 2022. Generative spoken dialogue language modeling.

Tu Anh Nguyen, Benjamin Muller, Bokai Yu, Marta R. Costa-jussa, Maha Elbayad, Sravya Popuri, Paul-Ambroise Duquenne, Robin Algayres, Ruslan Mavlyutov, Itai Gat, Gabriel Synnaeve, Juan Pino, Benoit Sagot, and Emmanuel Dupoux. 2024. Spirit-lm: Interleaved spoken and written language model.

Tu Anh Nguyen, Maureen de Seyssel, Patricia Rozé, Morgane Rivière, Evgeny Kharitonov, Alexei Baevski, Ewan Dunbar, and Emmanuel Dupoux. 2020. The zero resource speech benchmark 2021: Metrics and baselines for unsupervised spoken language modeling.

Vivian T Nguyen, Otto Versyp, Christopher Cox, and Riccardo Fusaroli. 2021. A systematic review and bayesian meta-analysis of the development of turn taking in adult-child vocal interactions. *Child development*.

Tsubasa Ochiai, Marc Delcroix, Yuma Koizumi, Hiroaki Ito, Keisuke Kinoshita, and Shoko Araki. 2020. Listen to what you want: Neural network-based universal sound selector. *arXiv preprint arXiv:2006.05712*.

Tsubasa Ochiai, Marc Delcroix, Yuma Koizumi, Hiroaki Ito, Keisuke Kinoshita, and Shoko Araki. 2020. Listen to What You Want: Neural Network-based Universal Sound Selector. In *arXiv*.

Yuki Okamoto, Shota Horiguchi, Masaaki Yamamoto, Keisuke Imoto, and Yohei Kawaguchi. 2022. Environmental sound extraction using onomatopoeic words. In *IEEE ICASSP*.

Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. 2016. Wavenet: A generative model for raw audio.

Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. 2016. Wavenet: A generative model for raw audio. In *arXiv*.

OpenAI. 2023. GPT-4 technical report. *CoRR*, abs/2303.08774.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback.

Tom Le Paine, Pooya Khorrami, Shiyu Chang, Yang Zhang, Prajit Ramachandran, Mark A. Hasegawa-Johnson, and Thomas S. Huang. 2016. Fast wavenet generation algorithm.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: an asr corpus based on public domain audio books.

Amy Pavel, Gabriel Reyes, and Jeffrey P. Bigham. 2020. Rescribe: Authoring and automatically editing audio descriptions. In *ACM UIST*.

Jay Prakash, Zhijian Yang, Yu-Lin Wei, Haitham Hassanieh, and Romit Roy Choudhury. 2020. EarSense: Earphones as a teeth activity sensor. In *MobiCom*.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. 2023. Robust speech recognition via large-scale weak supervision. In *International conference on machine learning*, pages 28492–28518. PMLR.

Andrew Reece, Gus Cooney, Peter Bull, Christine Chung, Bryn Dawson, Casey Fitzpatrick, Tamara Glazer, Dean Knox, Alex Liebscher, and Sebastian Marin. 2023. The candor corpus: Insights from a large multimodal dataset of naturalistic conversation. *Science Advances*, 9(13):eadf3197.

Resemble-Ai. 2019. Resemble-ai/resemblyzer: A python package to analyze and compare voices with deep learning.

Apple AirPods Max Wireless Headphones Review. 2023. `https://www.rtings.com/headphones/reviews/apple/airpods-max-wireless#page-test-results`.

Jonathan Le Roux, Scott Wisdom, Hakan Erdogan, and John R. Hershey. 2018. SDR - half-baked or well done?

Paul K. Rubenstein, Chulayuth Asawaroengchai, Duc Dung Nguyen, Ankur Bapna, Zalán Borsos, Félix de Chaumont Quitry, Peter Chen, Dalia El Badawy, Wei Han, Eugene Kharitonov, Hannah Muckenhirn, Dirk Padfield, James Qin, Danny Rozenberg, Tara Sainath, Johan Schalkwyk, Matt Sharifi, Michelle Tadmor Ramanovich, Marco Tagliasacchi, Alexandru Tudor, Mihajlo Velimirović, Damien Vincent, Jiahui Yu, Yongqiang Wang, Vicky Zayats, Neil Zeghidour, Yu Zhang, Zhishuai Zhang, Lukas Zilka, and Christian Frank. 2023. Audiopalm: A large language model that can speak and listen.

Steve Rubin, Floraine Berthouzoz, Gautham J. Mysore, Wilmot Li, and Maneesh Agrawala. 2013. Content-based tools for editing audio stories. In *ACM UIST*.

Justin Salamon, Duncan MacConnell, Mark Cartwright, Peter Li, and Juan Pablo Bello. 2017. Scaper: A library for soundscape synthesis and augmentation. In *WASPAA*.

Emanuel A Schegloff. 2000. Overlapping talk and the organization of turn-taking for conversation. *Language in society*, 29(1):1–63.

SDK. 2023. Steam audio. `https://valvesoftware.github.io/steam-audio/`.

ShanonPearce. 2022. Shanonpearce/ash-listening-set: A dataset of filters for headphone correction and binaural synthesis of spatial audio systems on headphones.

Sheng Shen, Nirupam Roy, Junfeng Guan, Haitham Hassanieh, and Romit Roy Choudhury. 2018. MUTE: Bringing IoT to noise cancellation. In *ACM SIGCOMM*.

Gabriel Skantze. 2021. Turn-taking in conversational systems and human-robot interaction: A review. *Comput. Speech Lang.*, 67:101178.

Tanya Stivers, Nick J. Enfield, Penelope Brown, Christina Englert, Makoto Hayashi, Trine Heinemann, Gertie Hoymann, Federico Rossano, Jan Peter De Ruiter, Kyung-Eun Yoon, Stephen C. Levinson, Paul Kay, and Krishna Y. 2009. Universals and cultural variation in turn-taking in conversation. *Proceedings of the National Academy of Sciences*, 106:10587 – 10592.

Michael A Stone and Brian CJ Moore. 1999. Tolerable hearing aid delays. i. estimation of limits imposed by the auditory path alone using simulated hearing losses. *Ear and Hearing*, 20(3):182–192.

Cem Subakan, Mirco Ravanelli, Samuele Cornell, Mirko Bronzi, and Jianyuan Zhong. 2021. Attention is all you need in speech separation. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 21–25. IEEE.

Cem Subakan, Mirco Ravanelli, Samuele Cornell, Frédéric Lepoutre, and François Grondin. 2022. Resource-efficient separation transformer. In *arXiv*.

Masahiro Sunohara, Chiho Haruta, and Nobutaka Ono. 2017. Low-latency real-time blind source separation for hearing aids based on time-domain implementation of online independent vector analysis with truncation of non-causal components. In *ICASSP*.

Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, et al. 2022. Lamda: Language models for dialog applications. *arXiv preprint arXiv:2201.08239*.

Noriyuki Tonami, Keisuke Imoto, Ryotaro Nagase, Yuki Okamoto, Takahiro Fukumori, and Yoichi Yamashita. 2022. Sound event detection guided by semantic contexts of scenes. In *arXiv*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288.

Efthymios Tzinis, Gordon Wichern, Aswin Subramanian, Paris Smaragdis, and Jonathan Le Roux. 2022. Heterogeneous target speech separation. *arXiv preprint arXiv:2204.03594*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *arXiv*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention is all you need.

Bandhav Veluri, Justin Chan, Malek Itani, Tuochao Chen, Takuya Yoshioka, and Shyamnath Gollakota. 2023a. Real-time target sound extraction. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE.

Bandhav Veluri, Justin Chan, Malek Itani, Tuochao Chen, Takuya Yoshioka, and Shyamnath Gollakota. 2023b. Real-time target sound extraction. In *IEEE ICASSP*.

Bandhav Veluri, Malek Itani, Justin Chan, Takuya Yoshioka, and Shyamnath Gollakota. 2023c. Semantic hearing: Programming acoustic scenes with binaural hearables. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, pages 1–15.

Bandhav Veluri, Malek Itani, Tuochao Chen, Takuya Yoshioka, and Shyamnath Gollakota. 2024a. Look once to hear: Target speech hearing with noisy examples. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, pages 1–16.

Bandhav Veluri, Benjamin N Peloquin, Bokai Yu, Hongyu Gong, and Shyamnath Gollakota. 2024b. Beyond turn-based interfaces: Synchronous llms as full-duplex dialogue agents. *arXiv preprint arXiv:2409.15594*.

Li Wan, Quan Wang, Alan Papir, and Ignacio Lopez Moreno. 2020. Generalized end-to-end loss for speaker verification.

Anran Wang, Maruchi Kim, Hao Zhang, and Shyamnath Gollakota. 2022a. Hybrid neural networks for on-device directional hearing.

Chengyi Wang, Sanyuan Chen, Yu Wu, Ziqiang Zhang, Long Zhou, Shujie Liu, Zhuo Chen, Yanqing Liu, Huaming Wang, Jinyu Li, Lei He, Sheng Zhao, and Furu Wei. 2023a. Neural codec language models are zero-shot text to speech synthesizers.

Quan Wang, Hannah Muckenhirn, Kevin Wilson, Prashant Sridhar, Zelin Wu, John Hershey, Rif A Saurous, Ron J Weiss, Ye Jia, and Ignacio Lopez Moreno. 2018. Voicefilter: Targeted voice separation by speaker-conditioned spectrogram masking. *arXiv preprint arXiv:1810.04826*.

Tianrui Wang, Long Zhou, Ziqiang Zhang, Yu Wu, Shujie Liu, Yashesh Gaur, Zhuo Chen, Jinyu Li, and Furu Wei. 2023b. Viola: Unified codec language models for speech recognition, synthesis, and translation. *CoRR*, abs/2305.16107.

Yuancheng Wang, Zeqian Ju, Xu Tan, Lei He, Zhizheng Wu, Jiang Bian, and Sheng Zhao. 2023c. Audit: Audio editing by following instructions with latent diffusion models.

Yuntao Wang, Jiexin Ding, Ishan Chatterjee, Farshid Salemi Parizi, Yuzhou Zhuang, Yukang Yan, Shwetak Patel, and Yuanchun Shi. 2022b. FaceOri: Tracking head position and orientation using ultrasonic ranging on earphones. In *ACM CHI*.

Zhong-Qiu Wang, Samuele Cornell, Shukjae Choi, Younglo Lee, Byeong-Yeol Kim, and Shinji Watanabe. 2023d. Tf-gridnet: Making time-frequency domain models great again for monaural speaker separation.

Gordon Wichern, Joe Antognini, Michael Flynn, Licheng Richard Zhu, Emmett McQuinn, Dwight Crow, Ethan Manilow, and Jonathan Le Roux. 2019. Wham!: Extending speech separation to noisy environments.

Scott Wisdom, John R Hershey, Kevin Wilson, Jeremy Thorpe, Michael Chinen, Brian Patton, and Rif A Saurous. 2019. Differentiable consistency constraints for improved deep speech enhancement. In *ICASSP*. IEEE.

Xudong Xu, Bo Dai, and Dahua Lin. 2019. Recursive visual sound separation using minus-plus net. In *IEEE/CVF ICCV*.

Xuhai Xu, Haitian Shi, Xin Yi, WenJia Liu, Yukang Yan, Yuanchun Shi, Alex Mariakakis, Jennifer Mankoff, and Anind K. Dey. 2020. Earbuddy: Enabling on-face interaction via wireless earbuds. In *CHI*.

Koji Yatani and Khai N. Truong. 2012. Bodyscope: A wearable acoustic sensor for activity recognition. In *UbiComp*.

Dong Yu, Morten Kolbæk, Zheng-Hua Tan, and Jesper Jensen. 2017. Permutation invariant training of deep models for speaker-independent multi-talker speech separation. In *ICASSP*. IEEE.

Amir Zadeh, Tianjun Ma, Soujanya Poria, and Louis-Philippe Morency. 2019. Wildmix dataset and spectro-temporal transformer model for monoaural audio source separation. *arXiv preprint arXiv:1911.09783*.

Dong Zhang, Shimin Li, Xin Zhang, Jun Zhan, Pengyu Wang, Yaqian Zhou, and Xipeng Qiu. 2023a. Speechgpt: Empowering large language models with intrinsic cross-modal conversational abilities. In *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, pages 15757–15773. Association for Computational Linguistics.

Haoning Zhang, Junwei Bao, Haipeng Sun, Youzheng Wu, Wenye Li, Shuguang Cui, and Xiaodong He. 2023b. Monet: Tackle state momentum via noise-enhanced training for dialogue state tracking.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. 2020. Dialogpt: Large-scale generative pre-training for conversational response generation. In *ACL, system demonstration*.

Xueliang Zhao, Wei Wu, Can Xu, Chongyang Tao, Dongyan Zhao, and Rui Yan. 2020. Knowledge-grounded dialogue generation with pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 3377–3390. Association for Computational Linguistics.

Katerina Zmolikova, Marc Delcroix, Tsubasa Ochiai, Keisuke Kinoshita, Jan Č ernocký, and Dong Yu. 2023. Neural target speech extraction: An overview. *IEEE Signal Processing Magazine*, 40(3):8–29.

Kateřina Žmolíková, Marc Delcroix, Keisuke Kinoshita, Tsubasa Ochiai, Tomohiro Nakatani, Lukáš Burget, and Jan Černocký. 2019. Speakerbeam: Speaker aware neural network for target speaker extraction in speech mixtures. *IEEE Journal of Selected Topics in Signal Processing*, 13(4):800–814.

Chapter A

SyncLLM training details

A.1 Hyperparameters

We trained SyncLLM with the Llama3-8b's original sequence length 8192. In the first stage, we train with a per-gpu batch size of 1 on 128 A100 GPUs, equivalent to a total batch of 8192 x 128 = 1M tokens. We use a learning rate of 3×10^{-5}, with 500 step warmup and train for 40k iterations. In the second stage, we reduce the batch size to 512k tokens, learning rate to 2.2×10^{-5} and warmup steps to 200, and train for 6000 iterations. In the last stage, we train with a batch size of 256k tokens, with a learning rate of 1.5×10^{-5} and 100 warmup steps, for 2000 iterations.

A.2 Benchmarking interleaving strategies

We explore two text-speech interleaving strategies in stage 1 of our training: i) Sentence-level interleaving: each sentence is chosen randomly to be either text modality or speech modality. ii) Turn-level interleaving: each turn is chosen randomly to be either text modality or speech modality, resulting in consistent modality for all the sentences within the turn. We compare them by evaluating on a set of spoken language understanding benchmarks proposed in Nguyen et al. [2020]. We report these results in Table A.1. On these tasks, we observe that sentence-level interleaving outperforms turn-level interleaving across all benchmarks.

Table A.1: Ablation evaluations over interleaving level. WUGGY, BLIMP, Topic-StoryCloze, and StoryCloze assess the knowledge and capacity of the model in lexical, syntactical, and semantic levels respectively. We report the accuracy based on negative-log-likelihood – normalized by the number of tokens – minimization prediction. The tasks are evaluated in the zero-shot setting.

Interleaving	WUGGY↑	BLIMP↑	Topic-StoryCloze↑	StoryCloze↑
Turn-level	63.0	56.0	76.5	55.1
Sentence-level	70.3	56.3	83.0	61.8

Table A.2: Comparison of average Pearson correlation of turn-taking event durations between generation and ground-truth continuation with SyncLLM in the two-model interaction setting. Measured on testsets comprising both Fisher and Candor testsets.

Latency	SyncLLM-F-F	SyncLLM-F-C
160 ms	0.32	0.36
200 ms	0.31	0.35
240 ms	0.28	0.32

Table A.3: Comparison of Pearson correlation of turn-taking event durations between prompt and generation.

Model	Fisher (in-distribution)				Candor (out-of-distribution)			
	ipu	pause	fto	Average	ipu	pause	fto	Average
dGSLM	0.60	0.34	0.23	0.39	0.43	0.20	0.09	0.24
SyncLLM-F (160 ms)	0.69	0.34	0.35	0.46	0.64	0.12	0.24	0.33
SyncLLM-F (200 ms)	0.57	0.49	0.29	0.45	0.61	0.34	0.13	0.36
SyncLLM-F (240 ms)	0.63	0.49	0.33	0.48	0.59	0.23	0.19	0.34
GT	0.72	0.53	0.31	0.52	0.54	0.30	0.12	0.32

A.3 Naturalness-MOS Instructions

Naturalistic turn-taking between two people is characterized by smooth transitions where each participant listens to the other, responds appropriately, and allows for pauses or silences, creating a balanced and dynamic interaction. Typically, the participants try to avoid overlapping speech, although this may occur especially when one participant provides information that they understood the other by using words like "yeah" or "uh-huh." Hesitations, pausing, silence, and repairs are also natural events that occur in a conversation between two people.

Here, you will listen to a dialogue between two people and provide a rating for how natural the turn-taking sounds regardless of its content (the meaning of the words used) and the clarity of voices.

Some of the samples are generated by an AI model, some are actual recordings of humans in conversation, and some are actual recordings of people, but with AI generated voices overlayed. Please try to assess

the naturalness of the turn-taking without taking into consideration the sound of the voices.

To begin, first listen to the "prompt" audio in its entirety. This is the first part of the conversation. Then listen to the "continuation" audio in its entirety. This is the second part of the conversation. Note that in many cases the voices in the prompt may differ from the voices in the continuation (including the perceived gender of the speakers). Your rating should reflect how natural the "continuation" audio sounds given the turn-taking characteristics you observe in the "prompt."

A.3.1 N-MOS & M-MOS

We provide the complete protocol used for human evaluation of turn-taking *Naturalness* and dialogue content *Meaningfulness*.

Audios presented

Please base your rating on the impression you have that two people are talking and listening naturally with one-another in the "continuation" audio.

- Excellent - basically indistinguishable from human-like turn-taking
- Good - minor differences from human-like turn-taking
- Fair - substantial differences from human-like turn-taking
- Poor - very little in common with human-like turn-taking
- Bad - essentially nothing in common with human-like turn-taking

Meaningfulness-MOS

In this task you will listen to a dialogue between two people and provide a rating for how meaningful their conversation is. By meaningful we mean the degree to which the content of the conversation is coherent and plausible (can you understand the intent of the speakers and does it sound like something people would reasonably talk about). Just as in everyday conversations, the content may or may not be perfectly grammatical, but must be understandable in the context of the conversation.

To begin, first listen to the "prompt" audio in its entirety. This is the first part of the conversation. Then listen to the "continuation" audio in its entirety. This is the second part of the conversation. Note that in many cases the voices in the prompt may differ from the voices in the continuation (including the perceived

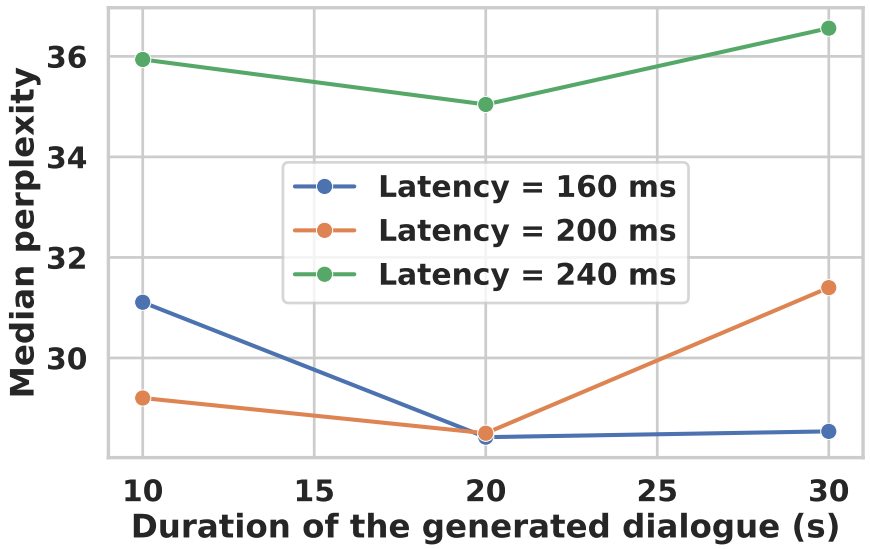


Figure A.1: Effect of latency on two-model interaction.

gender of the speakers). Your rating should reflect how meaningful the "continuation" audio is, given the "prompt."

Audios presented

Please base your rating on the impression you have that the continuation is a meaningful "continuation" of the prompt audio - that it represents a plausible direction the conversation would go and is coherent.

- Excellent - all of the conversation content is plausible and coherent
- Good - most of the conversation content is plausible and coherent
- Fair - some of the conversation content is plausible and coherent
- Poor - little of the conversation content is plausible and coherent
- Bad - basically none of the conversation content is plausible and coherent

A.3.2 List of all keywords

all_keywords = 'dancing', 'nutrition', 'motorcycles', 'minimalism', 'crafts', 'makeup', 'cars', 'singing', 'wine', 'candy', 'backpacking', 'nature', 'television', 'fitness', 'museums', 'yoga', 'skincare', 'travel', 'guitar', 'beer', 'film', 'skiing', 'coffee', 'theater', 'theme_parks', 'piano', 'restaurants', 'trains', 'gardening', 'books', 'violin', 'football', 'programming', 'water', 'developer', 'concerts', 'health', 'baking', 'mindfulness', 'knitting', 'climate', 'hiking', 'cooking', 'podcasts', 'tea', 'student', 'art', 'sunshine', 'camping',

'photography', 'reading', 'snacks', 'history', 'bowling', 'VR', 'exercise', 'gaming', 'woodworking', 'music', 'food', 'festivals', 'surfing', 'bridges', 'shopping', 'movies', 'graffiti', 'ice skating', 'sports', 'animals', 'drawing', 'fashion', 'ocean', 'soccer', 'skating', 'basketball', 'running', 'climbing', 'welding', 'sleep', 'anime', 'tennis', 'religion', 'office', 'drums', 'philosophy', 'dance', 'DIY', 'volleyball', 'beach', 'social_media', 'writing', 'museum', 'comics', 'driving', 'meditation', 'swimming', 'cricket', 'psychology', 'pets', 'painting'

A.4 Effect of latency on full-duplex interaction

In Fig. A.1, we compare the performance in the interaction setting with different latencies. We find that our method is robust to a latency as much as 200 ms, but the performance drops with latency greater than that. Similar to our naturalness evaluation in the continuation setting in §3.5.2, to evaluate turn-taking capability of SyncLLM in interaction setting, we compare Pearson correlation of the duration of turn-taking events in generation and ground-truth continuations. In Table A.2, we observe that on a combined test set of in-distribution and out-of-distribution prompts, performance in the interaction setting closely matches with latencies 160 ms and 200 ms, but drops with 240 ms.

A.5 Turn-taking event correlation between prompt and generation

Similar to the naturalness evaluation in Table 3.2, where we consider ground-truth continuation as the reference for turn-taking event statistics, we could also consider prompt as the reference. In a way, this measures style consistency between prompt and the continuation. In Table A.3, we compare turn-taking event correlation of generations of our method in continuation setting, with that of dGSLM method. We observed that our method demonstrates better turn-taking correlation with the prompts as well.